前言

计算机和万维网（World Wide Web）正在迅速而显著地改变着生物学研究的面貌。现在，“模式转变(paradigm shift)”这个词可以来描述一切事物 —— 从新的经济趋势到新口味的可乐。但是，生物科学的模式转变还是传统意义上的。理论和计算生物学已经在生物科学的“边缘”中存在了几十年。但就在短短的几年间，基因组研究中大量新的生物学数据和分析这些基因组数据所必需的计算机应用开始影响到生物学的每一个方面。过去科学家要在实验室进行的研究，现在可以在计算机上进行，因为需要查找数据库信息，从中提出新的假说。

在过去的 20年中，PC机和超级计算机已经走进了各学科科学家的生活。一开始，PC机是作为实际计算能力很小的昂贵的新生事物出现的，现在它已经具有 10年前超级计算机的计算能力。就像计算机已经代替了写作者的打字机和会计的分类账目一样，在实验室中，它们在从实验室仪器中控制和收集资料方面发挥着作用，作为一种存储资料的方法，计算机已有能力完全代替实验室记录和文件。计算机数据库存储的数据比其他非电子形式的记录要容易访问。除了在数据的存储、分析和可视化方面的作用以外，在对任何可用数学方式进行描述的系统的理解上，计算机的功能也非常强大，因而出现了计算生物学这一学科，更新的叫法是生物信息学。

生物信息学(bioinformatics）是信息技术在生物数据处理上的应用，它是一门快速发展的学科。在过去的 20年中，将生物数据存储在公共的数据库中变得越来越普遍了，而且这些数据库也在呈指数增长。生物学的文献也在呈指数增长。如果没有计

VISIT…

LANZAROTE Caliente.COM

算机工具的帮助，即使是最热情的研究工作者，想要得到相关领域内最新的必要信息也是不可能的。通过 Web，用户在任何地方都可以与任何站点上的数据和资料进行交互 —— 只要他们知道怎样建立正确的工具。

生物信息学首先是生物科学，但是它又不仅仅是生物科学。它往往不是致力于研究精致的算法，而是解决实际中的问题。生物信息学家是工具建立者，为了建立有用的工具，对于他们来说，对生物学问题的理解和对计算方法的理解同等重要。生物信息学算法需要涉及复杂的科学推测，这些推测以独特的方式使程序开发和数据建模变得更加复杂。

生物信息学和计算生物学的研究范围包罗万象：从将生物系统中提取出的特性转变为数学或物理模型，到实现数据分析的新算法，再到开发数据库和 Web工具来访问它们。为了从事计算方面的研究，生物学家必须熟练应用在各种操作系统中运行的多种软件工具。本书介绍并解释了在生物信息学研究中最常用的工具。本书包括许多附加信息和背景资料，以帮助理解如何很好地利用这些工具，并解释它们的重要性。我们希望本书能帮助你迈出在研究工作中高效应用计算机的第一步。

本书的读者

大多数生物科学的学生和研究人员应用计算机的目的越来越不仅限于进行文字处理或数据收集和作图。但是，许多人没有计算机科学或计算理论的背景，对他们而言，计算生物学和生物信息学领域可能非常广泛而复杂。本书是受我们的学生和同事的启发而撰写的，它绝不是涵盖了生物信息学各方面的圣经。然而，本书对生物信息学一些最重要的主题的介绍却是非常深入的。为了找到生物序列、基因组和分子结构数据库的信息，我们介绍了标准的计算技术；我们讨论了如何鉴定基因并检测可辨识基因家族的特征模式；讨论了系统发育关系、分子结构和生物化学特性建模。我们还讨论了如何将计算机作为一个工具来组织数据，系统地思考数据分析过程，并开始思考数据处理的自动化。

生物信息学是相当前沿的，因此即使看懂一本像这样介绍性的书也需要有一定的背景知识。为了很好地理解本书，应该有分子生物学、化学和数学的理论知识或学习经历。在计算机处理方面修一门或两门大学课程也是很有帮助的。

本书的结构

我们精心地安排了书中的材料，既可以从头到尾阅读，也可以跳过前面的章节学习后面的内容。本书分为四部分：

第一部分，概述

第一章，计算机时代的生物学，对生物信息学这门学科进行定义，讲述一些有关的历史，简要介绍本书包含的内容以及为何包含这些内容。

第二章，生物学问题的计算方法，介绍生物信息学和分子生物学的中心概念，以及产生大量生物学数据的技术和研究动机，还包括每个生物学家都需要知道的不断增长的基本计算机程序列表。

第二部分，生物信息学工作站

第三章，建立工作站，介绍 Unix、然后是在 PC机上安装 Linux的基础知识，安装并运行软件。

第四章，Unix 中的文件和目录，全面讲述了 Unix 文件系统，包括文件层次、命名模式、常用的目录命令和多用户环境下的工作。

第五章，在 Unix 系统下工作，解释用户日常最可能遇到的 Unix 命令，包括从文件中阅读、编辑和提取信息；正则表达式； shell 脚本；与其他计算机的信息交流。

第三部分，生物信息学工具

第六章，在 Web 上进行生物学研究，讲述如何在 Web 上寻找信息的艺术。本章包括搜索引擎和搜索过程，到哪里去寻找科学文献和软件，怎样应用在线信息资源以及公共生物学数据库。

第七章，序列分析、成对比对以及数据库搜索，本章一开始对分子进化进行了回顾，然后介绍了成对（pairwise）序列分析技术的基础，比如：预测基因的位置，整体和局部的比对(alignment)，应用 BLAST和 FASTA 以局部的比对为基础在数据库中进行查找。本章包括用于序列分析的多功能工具。

第八章，多序列比对、进化树和简图，开始学习一系列相关的基因和蛋白质。它包含用 ClustalW 和 Jalview工具进行多序列比对的策略，然后讨论用于系统发育的分析、构建简图（ profile）和基序的工具。

第九章，蛋白质结构的可视化和结构性质的计算，包括蛋白质的三维分析和用来计算蛋白质结构性质的工具。本章一开始回顾了蛋白质化学的历史，然后转到以 Web 为基础的蛋白质结构分析工具的讨论；结构分类、比对和分析；溶剂的可接近性以及溶剂的相互作用；计算蛋白质的物理化学性质。最后以结构优化及浏览蛋白质结构数据库结束本章内容。

第十章，根据序列预测蛋白质的结构和功能，包括根据蛋白质的序列确定蛋白质结构的工具。本章讨论蛋白质序列的特征检测，二级结构预测以及三维结构预测。最后举例说明蛋白质建模。

第十一章，基因组学和蛋白质组学工具，将它们放在一起进行分析。至今为止，我们已经讲述了分析单一序列或结构的工具和技术，以及单基因长度的多序列比较的工具和技术。本章讨论在研究基因组中所有基因的整合功能时可用到的一些数据类型和工具，包括两个完整的基因组的序列测定，在 Web 上访问基因组的信息，注释和分析整个基因组序列以及新出现的技术以及蛋白质组学。

第四部分 数据库和可视化

第十二章，用 Perl 进行数据自动化分析，告诉大家怎样应用 Perl语言，从大量的数据中提取出想得到的信息。本章目的不是教大家如何用 Perl 语言进行编程，但是在本章对Perl语言进行了一些简单的介绍。并举例以帮助大家用自己的方式学习编程。

第十三章，构建生物学数据库，介绍数据库的概念。包括在生物学研究中的数据库的类型，构建生物学数据库的数据库软件，数据库语言（尤其是 SQL语言）以及正在发展的与数据库交互的基于 Web的软件。

第十四章，可视化和数据采集，包括分析实验结果的计算工具和技术。本章的第一部分介绍在生物信息学研究中发展起来用于使数据可视的程序。其范围从一般意义上的线图和用于大量数据的统计包，如 Grace和 gnuplot，到以可解释的形式来提交序列和结构信息的程序，如 TEXshade。本章第二部分讲述在生物信息学应用中现有的数据采集工具 —— 在大量数据中寻找、解释、求取模式的过程。

研究生物信息学的方法

我们承认，我们是结构生物学家（实际上是生物物理学家）。对我们来讲，不考虑基因的蛋白质产物而研究基因是非常困难的。DNA序列对我们来说不仅仅意味着序列本身。对结构生物学家来说，基因（只有少数例外）意味着三维结构、分子形状和构象的变化、活性位点、化学反应和细微的分子内相互作用。在本书中，我们着眼于结构生物学家和生化学家以理解生物学功能的化学基础可能使用的序列信息。也许本书忽略了对分子生物学家和遗传学家来说很重要的一些序列分析的应用，所以欢迎读者发表自己的意见。

本书引用的 URL

要了解在本书中引用的URL和作为生物信息学附加材料的更多信息，可参看本书的网页，列在“建议与评论”部分。

本书排版约定

本书英文使用以下字体约定：

斜体（*Italic*）

用于命令、文件名、目录、变量以及 URL。

等宽字体（`Constant width`）

用于代码实例以及显示命令的输出。

等宽斜体(*`Constant width italic`*)

用于“用法”短语表示变量。

建议与评论

本书的内容都经过测试，尽管我们做了最大的努力，但错误和疏忽仍然是在所难免的。如果你发现有什么错误，或者是对将来的版本有什么建议，请通过下面的地址告诉我们：

美国：

O'Reilly & Associates,Inc.
101 Morris Street
Sebastopol, CA 95472

中国

100080 **北京市海淀区知春路 49号希格玛公寓** B **座** 809 **室**
奥莱理软件（北京）有限公司

本书有一个网页，上面附有勘误表、实例和其他附加信息。可以访问以下网页：

http://www.oreilly com/catalog/bioskills/

如评论或探讨技术问题，可发 email**至：**

bookquestions@oreilly.com
info@mail.oreilly.com.cn

要查找有关本书更多的内容、评论、软件、资源中心以及 O'Reilly **网络，请访问我们的网站：**

http://www.oreilly.com
http://www.oreilly.com.cn

致谢

Cynthia**的致谢：首先感谢所有听到我说“这本书就要写完了”而没有嘲笑我的人，因为这句话我说了有一千次了。感谢我的家人和朋友，因为在最近几个月中，我很少给他们打电话，但是他们能容忍我；感谢在** 2000**年秋季修生物信息学课程的学生，因为他们在我第一次生物信息学实验中积极参与，帮我确定了哪些主题需要更系统地解释；感谢我在弗吉尼亚理工学院的同事，因为在一年来，就有关什么是生物信息学以及进行生物信息学学习的学生应该学习什么这些问题，我与他们进行了有趣的讨论；感谢我的朋友和同事** Jim Fenton **，因为他在本书形成的早期做出了不少贡献；感谢我的论文导师** Shankar Subramaniam**。我也要感谢我们的技术审校，**

Sean Eddy、Peter Leopold、Andrew Odewahn、Clay Shirky 和 Jim Tisdall，因为他们对本书提出了建设性的评论和非常棒的建议。最后，感谢 O'Reilly的工作人员和本书的编辑 Lorrie LeJeune，感谢他们在本书写作过程中付出的耐心和帮助。

Per 的致谢：首先感谢我的导师， Shankar Subramaniam 教授，他总有源源不断的灵感，而且是我们在 UCSD （加大圣迭戈分校）实验室的良好的工作氛围的支柱。感谢我的两位导师：UCSD的教授 Charles Elkan，以及现在华盛顿大学的 Michael R. Brent教授，他们充满智慧的指导使我对计算问题的理解得以成形。 Sanna Herrgard 和 Markus Herrgard阅读了本书的前几稿并提供了有价值的评论和精神上的支持。本书也得益于来自 Ewan Birney、Phil Bourne、Jim Fenton、Mike Farnum、Brian Saunders 和 Winny Tan 的建议和富于帮助的谈话。感谢 O'Reilly的 Joe Johnston提供了第十二章中有关 Perl 的建议和代码。我们的技术审校提出了不可缺少的建议和贡献，在此我特别感谢 Sean Eddy、Peter Leopold、Andrew Odewahn、Clay Shirky 和 Jim Tisdall，他们对本书的工作非常仔细认真。与 O'Reilly 的工作人员一起合作非常愉快，特别是编辑 Lorrie LeJeune，他耐心而又快乐地指导我们完成本书的写作。最后，如果没有家庭的支持和鼓励，我也不可能完成本书的写作，因此感谢我的家人。

第一部分

概述

第一章，计算机时代的生物学

第二章，生物学问题的计算方法

第一章

计算机时代的生物学

从物种和群体的相互影响，到生物体内组织和细胞的功能研究，都是生物学研究的内容。一般认为生物学是对有生命的事物所进行的研究。在此研究过程中，生物学家首先要收集资料，然后对这些资料进行解释。现在已经是 21世纪，我们已经可以应用比较复杂的实验室技术进行课题的研究，因此收集实验资料的速度比对它进行解释的速度要快得多。比如：我们拥有大量的 DNA 序列资料。但是，如何才能计算出是DNA 的哪一部分控制生命的各种化学过程呢？我们了解一些蛋白质的结构和功能，但是如何才能确定一个新蛋白质的功能？如何根据蛋白质的序列信息预测它的形状？我们知道可将DNA 翻译成蛋白质的简单密码，但是如何发现密码中新的代码并将它们加入到 DNA- 蛋白质字典中去呢？

生物信息学（bioinformatics）是应用信息来理解生物学的科学；我们可以用生物信息学来回答上述问题以及其他类似的问题。遗憾的是，由于对人类基因组图谱研究的大肆炒作，生物信息学的定义非常不统一，生物信息学一词的意义因使用者不同，使用的方式也五花八门。严格地说，生物信息学是广泛意义上的计算生物学（computational biology）的一门分支，它在为生物学系统建模中应用了量化分析技术。本书既会从生物信息学到计算生物学进行阐述，又会反过来从计算生物学到生物信息学进行阐述。本书的目的不在于说明二者之间的区别，而在于讲述一系列对于今天的分子生物学家理解和应用基本计算工具非常重要的工具和技术。

生物信息学的范围很大程度上依赖于专家们在统计学方法和模式识别方面的工作。进行生物信息学研究的工作者来自许多领域，包括数学、计算机科学和语言学等领

域。遗憾的是，生物学是既特殊又普通的学科。对于那些不能完全理解生物学数据的出处及其含义的人来说，在生物信息学的研究工作中寻找规律、进行预测非常困难。如果提供了算法、数据库、用户界面以及统计学工具，生物信息学有可能进行许多激动人心的工作，比较 DNA 序列，生成有潜在意义的结果。这里的“潜在意义”是一个非常重要的词语。这些新的工具也可以给你提供机会，可以进一步解释数据并赋予其以前未知的含义。我们不能夸大理解这些工具局限性的重要性。但是一旦理解了生物信息学的内容并且能够很聪明地应用生物信息学系统，你的研究速度将会非常惊人。

计算是如何改变生物学的？

有机体的遗传和功能信息由 DNA、RNA 及蛋白质存储，它们是由小分子组成的线性结构。这些大分子是由那些已经众所周知的化合物组成的： DNA 由四种脱氧核苷酸（腺嘌呤、胸腺嘧啶、胞嘧啶和鸟嘌呤）组成， RNA 由四种核苷酸（腺嘌呤、尿嘧啶、胞嘧啶和鸟嘌呤）组成，蛋白质由 20种氨基酸组成，由于这些大分子是由确定的组分组成的线性结构，它们可以用符号的序列来代表。这些序列可通过比较来寻找它们之间的相似性，而这些相似性可能暗示着分子之间形式或功能上是相关的。

序列比较可能是分子生物学家最常用的计算工具。万维网可以使一个基因组序列数据的公共数据库能够通过同样的界面为世界范围的用户提供服务。用一个常用的称作 fsBLAST 的计算机程序，生物学家可以将一个不典型的 DNA 序列与收集到的全部的DNA序列进行比较。在下面的一节，我们将举一个例子来解释如何应用BLAST程序进行序列比较，帮助我们理解一种疾病。

蝇的复眼

果蝇（drosophila melanogaster）是研究动物从胚胎到成年发育阶段的常用模型系统。果蝇有一种称为“eyeless（无眼）”的基因，如果将此基因敲除(knock out，也就是说，应用分子生物学方法，将其从基因组中去除），将会导致果蝇失去眼睛。很明显，*eyeless* 这个基因在眼睛的发育过程中起着重要作用。

研究者已经鉴定出一个与“aniridia（无虹膜）”现象有关的人类基因。如果人类缺

少这个基因（或对此基因进行基因突变，使其蛋白质产物不能正确表达功能），眼睛形成以后将没有虹膜。

如果将基因“aniridia”插入到因“基因敲除”而没有眼睛的果蝇中，果蝇的眼睛可以正常形成，这种巧合非常有趣。是否即使果蝇和人是两种非常不同的生物，*eyeless* 和 *aniridia* 的功能也具有相似性？这很有可能。为了仔细研究 *eyeless* 和 *aniridia* 共同作用的原理，我们要对比它们的序列。然而，一定要记住的是，基因间的相互作用是非常复杂的，为了得到一个较为确切的答案需要非常仔细地进行实验。

15年之前，如果要寻找 *eyeless* 和 *aniridia* DNA 序列的相似性，其困难程度不亚于大海捞针。许多科学家通过手工排序来比较各自的基因序列，将一个序列放到另一个序列的下面，按照字母顺序逐一寻找其匹配情况。这种工作非常浪费时间，而且很容易引起眼睛疲劳。

在20世纪80年代末，可以快速比较序列的计算机程序永久地改变了分子生物学。生物学序列的成对比较是大多数广泛应用的生物信息学技术的基础。许多应用于生物学的工具，比如多序列比对、系统发育分析、基序的鉴定、同源性建模的软件；以Web 为基础的数据库搜索服务都依赖这个成对序列－比较算法作为它们功能的核心部分。

近来生物学家运用序列－比对程序，如BLAST 和 FASTA，可以在很短的时间内找出大量相匹配的序列。这些程序的应用是如此普遍，以至于你在第一次应用生物信息学工具和生物学数据库时，就可能是通过NCBI(National Center for Biotechnology Information，美国国家生物技术信息中心）BLAST 的 Web界面，图 1-1是在为进行 BLAST 查找时，向 NCBI 提交数据的表单。

基因序列的标签

在用 BLAST比较 *eyeless*和 *aniridia*的序列之前，首先要了解序列比对的基础知识。

很重要的一点是，生物学的序列（DNA或蛋白质）具有化学的功能，但是当简化为单字母代码时，它就可以作为一个惟一的标签，好像是一个条形码。从信息技术的角度来看，序列的信息是无价的。序列标签可应用于基因、基因的产物、功能及其在细胞代谢中的作用等等。寻找与特定基因相关的信息的用户，可以用成对的序列比较来获得有关序列标签的信息。

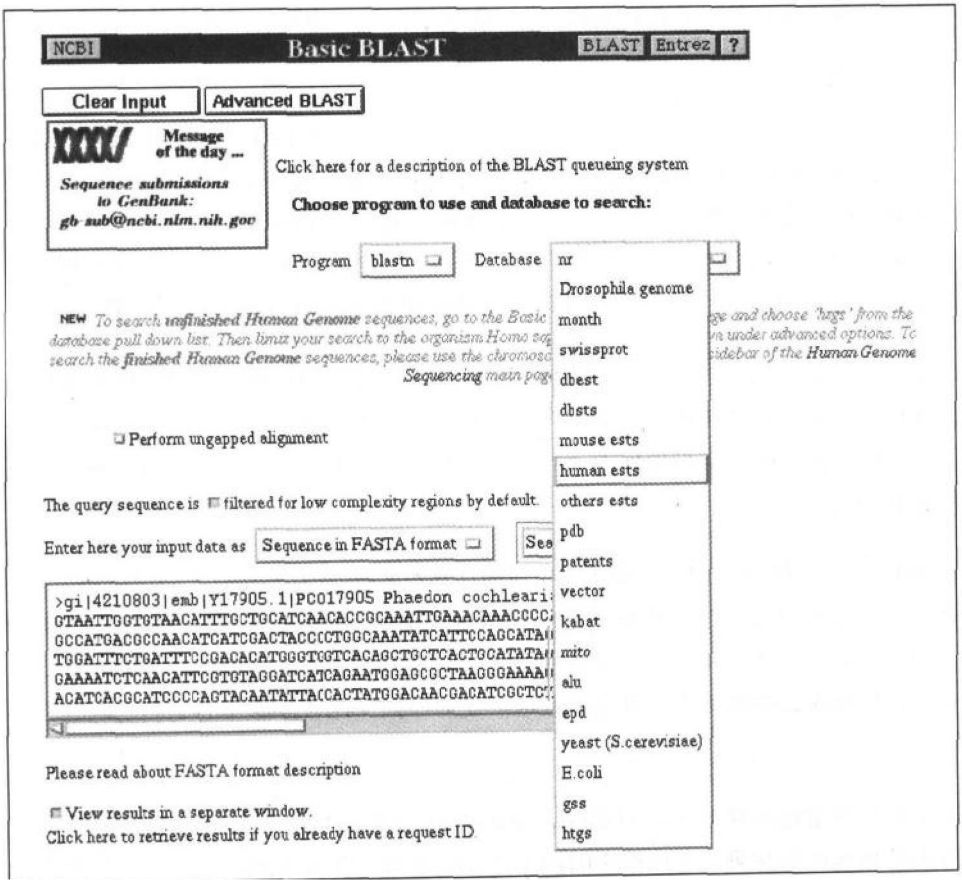


图1-1　在NCBI的核苷酸数据库中提交一个BLAST搜索时的表单

然而，对于这些序列的标签来说，最重要的并不只是标识一个特别的基因。它们也包含有意义的生物学模式，允许用户比较不同的标签，连接信息并进行推理。因此，这些标签不仅可连接某个基因的所有信息，而且可帮助用户连接与此基因有或微小或显著差异的基因序列所带有的信息。

如果简单的标签对于弄懂生物学数据的含义已经足够，那么就可以输入一个惟一的数字（也就是说，一个GenBank ID）至每一个DNA序列，然后再进行处理。但是生物学序列和进化是相关的，所以在两个序列标签中有部分相匹配是很常见的。BLAST和简单的关键词查找的区别在于，BLAST具有检测与整个蛋白质序列部分匹配的能力。

用 BLAST 比较 eyeless 和 aniridia

如果用 BLAST 比较 *eyeless* 和 *aniridia* 序列，就会发现 *eyeless* 部分匹配 *aniridia*。下面是进行 BLAST 搜索的原始资料：

```
pir||A41644 homeotic protein aniridia - human
       Length = 447

  Score = 256 bits (647), Expect = 5e-67
  Identities = 128/146 (87%), Positives = 134/146 (91%), Gaps = 1/146 (0%)

 Query: 24  IERLPSLEDMAHKGHSGVNQLGGVFVGGRPLPDSTRQKIVELAHSGARPCDISRILQVSN 83
            I R P+  M + HSGVNQLGGVFV GRPLPDSTRQKIVELAHSGARPCDISRILQVSN
 Sbjct: 17  IPRPPARASMQNS-HSGVNQLGGVFVNGRPLPDSTRQKIVELAHSGARPCDISRILQVSN 75

 Query: 84 GCVSKILGRYYETGSIRPRAIGGSKPRVATAEVVSKISQYKRECPSIFAWEIRDRLLQEN 143
           GCVSKILGRYYETGSIRPRAIGGSKPRVAT EVVSKI+QYKRECPSIFAWEIRDRLL E
 Sbjct: 76 GCVSKILGRYYETGSIRPRAIGGSKPRVATPEVVSKIAQYKRECPSIFAWEIRDRLLSEG 135

 Query: 144 VCTNDNIPSVSSINRVLRNLAAQKEQ 169
            VCTNDNIPSVSSINRVLRNLA++K+Q
 Sbjct: 136 VCTNDNIPSVSSINRVLRNLASEKQQ 161

  Score = 142 bits (354), Expect = 1e-32
  Identities = 68/80 (85%), Positives = 74/80 (92%)

 Query: 398 TEDDQARLILKRKLQRNRTSFTNDQIDSLEKEFERTHYPDVFARERLAGKIGLPEARIQV 457
            +++ Q RL LKRKLQRNRTSFT +QI++LEKEFERTHYPDVFARERLA KI LPEARIQV
 Sbjct: 222 SDEAQMRLQLKRKLQRNRTSFTQEQIEALEKEFERTHYPDVFARERLAAKIDLPEARIQV 281

 Query: 458 WFSNRRAKWRREEKLRNQRR 477
            WFSNRRAKWRREEKLRNQRR
 Sbjct: 282 WFSNRRAKWRREEKLRNQRR 301
```

输出结果显示了 *eyeless* 和 *aniridia* 基因的蛋白质序列中两个高分值（high-scoring）匹配区域的局部比对结果。在一组由三行组成的查询序列（提交给 BLAST 服务器的 *eyeless* 序列）中，最上面一行是由 BLAST 服务器输出的 *eyeless* 序列，底部一行是 *aniridia* 序列，中间一行是两个序列匹配的部位。如果在中间行有字母，则表示序列匹配的确切位置。如果在中间行有 + 号，表示在此位置两个序列不相同，但是在这些氨基酸之间具有化学结构的相似性（如 D 和 E，天冬氨酸和谷氨酸）。如果中间行中的位置是空格，表示在此位置两个序列不匹配。

从本例中可以看到，如果提交整个 *eyeless* 基因序列（就像使用标准的关键字搜索一样）并寻找确切的匹配，你将得不到任何结果。局部的序列区域是完整蛋白质的一

部分：在 *eyeless* 中 24 ~ 169 区段和人的 *aniridia* 基因的 17 ~ 161 区段匹配，*eyeless* 的 398 ~ 477 区段和 *aniridia* 的 222 ~ 301 区段匹配。而序列的其他区段不匹配！即使在显示出的两个区域中，虽然它们是匹配的，但也不是 100% 的匹配。

然而，这种部分匹配是非常有意义的。它告诉我们不太为人所知的人类的 *aniridia* 基因与果蝇的 *eyeless* 基因的序列具有一定的相关性。我们已经对果蝇的 *eyeless* 基因了解了很多，从结构和功能（它是一个可提高其他基因活性的 DNA 结合蛋白），到它对果蝇表现型（phenotype，成年果蝇的外形）的影响。

BLAST可发现整体不完全匹配的成对序列中局部匹配的区域。可以将匹配延伸到序列中的单字符不同以外。它可以在各个方向上延伸其匹配，直到序列匹配的整体分值非常低。结果，BLAST可以检测到序列之间不能精确复制的模式，因此，序列之间亲缘关系较远的，其匹配不太精确的情况也具有生物学意义。

根据两个标签匹配的程度不同，可以将一个标签的信息转移到另一个标签。如果两个全长序列的匹配程度较高，则提示功能具有相似性，虽然这种鉴定是推论性质的，其结果还需要通过实验来证实。在有关 *eyeless* 和 *aniridia* 基因的例子中，科学家希望通过研究果蝇眼睛发育过程中 *eyeless* 基因的作用，帮助人们理解在人类眼睛的发育过程中 *aniridia* 所起的作用。

生物信息学难道仅仅是建立数据库吗？

在生物信息学中，我们现在考虑得较多的部分——序列比较、序列数据库搜索、序列分析——比仅仅设计和建立数据库要复杂得多。生物信息学家（或计算生物学家）不仅仅是捕获、处理和呈现出数据，还包括从统计学、物理学、计算机科学和工程学等各种定量领域中获得灵感。图 1-2 显示了定量学科与生物学在各个层次上的交叉，从分析序列数据和蛋白质结构，到建立代谢模型以及种群和生态分析等的定量分析。

生物信息学首先是生物科学的分支，而且是生物科学的重要分支。生物信息学的主要目的不是发展最精致的算法或最神秘的分析；其目的是发现生物体以怎样的方式生存。像已经大大扩展的生物学家已经掌握的分子生物学方法一样，生物信息学是一个工具，其本身不是目的。生物信息学家是工具的建立者，为了建立有用的工具，他们必须理解生物学的问题和计算方法。

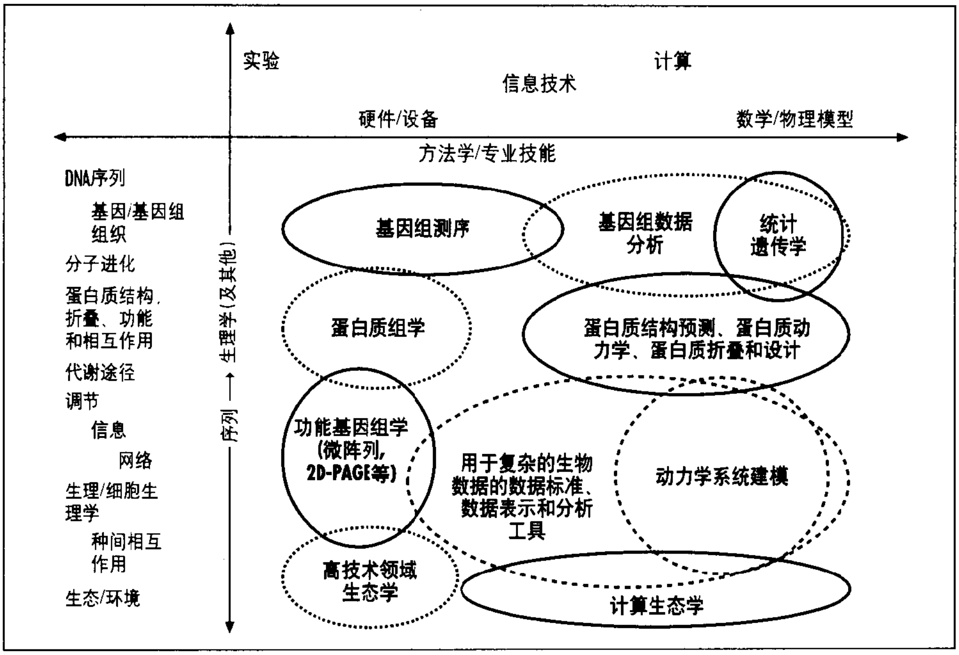


图 1-2　技术怎样和生物学相互交叉

生物信息学和计算生物学研究包括从将生物系统的性质抽象出为数学或物理模型，到实现数据分析的新算法，以及开发数据库和访问数据库的 Web 工具。

生物学最初的信息时代

生物学作为一门特殊的学科，意味着生物学家不仅需要记住其细节，还要清楚一般的原则。从 17 世纪开始，生物学家已经开始着手解决信息管理的问题了。

进化概念来源于早期生物学家对生物种类进行的编目及比较工作。近三个世纪以来，生物学家对各生物种类进行编目，开始是对动物和植物，以后由于复合显微镜的应用，也开始对显微生物进行编目。迄今为止还在不断发现新的生命形式和以前所未知的化石中保存的已经灭绝的生命形式。

在当时植物和动物目录导致大量信息的产生。在 16 世纪中期，Otto Brunfels 发表了第一部描述植物分类的重要的现代著作，《Herbarium vitae eicones》。由于欧洲

人游历渐广，进行编目的物种随之增加，植物园和植物标本集开始出现。在亚里士多德的学生 Theophrastus生存的时代，进行编目的植物类型的数目有500种。到 1623年，Casper Bauhin 已经发现 6000种植物。不久，John Ray 引入了区分动物和植物物种的概念，形成了以解剖形状为基础区分不同种类的指导路线。在 18 世纪 30年代，Carolus Linnaeus 对 18000 种植物和 4000 多种动物进行分类，建立了现代分类学命名系统界、纲、属、种的基础。 18 世纪末期，Baron Cuvier 列出了 50000 多种植物。

对于这个探索和分类的时期，生物学家的当务之急是将物种纳入到正规的分类学中，他们对此的意见不大相同。一本植物学教科书可能包含好几卷植物资料，含有对于每个种类的植物进行的图示和描述。生物学家需要面对的问题，是怎样组织、得到并合理地增加这些信息。很明显，任何粗心的观察者都可以发现，一些生物特别相似，而其他的生物却不太相似。比如大鼠和小鼠比较相似，其相似程度与小鼠和狗相比要强得多。但是如果生物学家不去查阅那几本有物种分类图的书，他怎样才能知道大鼠和小鼠较相似呢（而大鼠又不是小鼠的别名）？需要创造出对每一种生物进行惟一的标识，总结它与其他生物之间的关系，并将所有的东西用很少的文字去概括的命名法（ nomenclature）。

相对来说，解决方法比较简单，但在当时是一个很大的革新。为增强其特异性，各物种是用一系列特征性不断加强的单字符名字进行命名而完成的，首先是最基本的分类：动物或植物？该生物属于的界。然后，随着专一性的提高，一直到纲、属、种。物种分类的图解方式如图 1-3 所示，现在被称为“ 生命之树 ”。

在地球上存在着上百万个物种的现代分类，即使对于最有热情的生物学家来说，记忆它们也是太复杂了，幸运的是，计算机现在提供了一种保存及访问物种分类的方法。亚利桑那大学的生命之树计划和 NCBI 的分类数据库是在线分类学课题的两个例证。

分类学是生物学遇到的第一个信息学问题。通过个体基因信息的收集和分类，现在生物学家又达到相似的情形。在基因水平上组织这些信息并让科学界分享这些知识的问题，不能通过提出一个命名法来解决。这从一开始就应该用计算机和数据库的知识来解决。

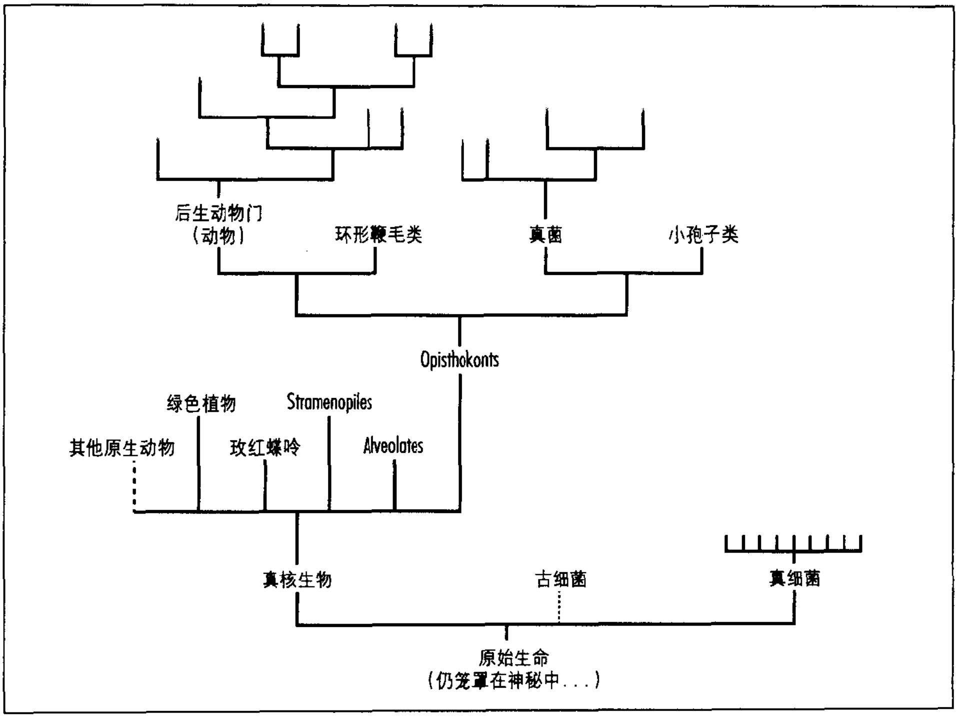


图 1-3　“生命之树”代表对物种分类的命名系统

在近半个世纪以来，计算机的发展和物理科学争相发展，使我们能够更细致地观察生物系统。图 1-4 显示了在近 20 年内生物学知识在以令人吃惊的速率增长。

如何在所得到的海量信息中发现有用的信息，本身便是一个有意义的研究课题。即使在 20 世纪 80 年代晚期，发现一个序列的数据库匹配结果都要用 5 页纸打印，而现在，这已成为一个例程。但是随着查找序列和结构数据库能力的增长，还会有其他的问题出现。这些问题对生物信息学领域是一种推动力。

信息学对生物学家意味着什么？

信息科学与表示、组织、处理、发布、维护和应用（特别是数字形式的）信息相关联。信息学和生物学交叉学科——生物信息学的实际含义不只有一种解释，很有可能你申请到了一个进行生物信息学研究的工作，却发现其工作前景与你想像的相差很远。

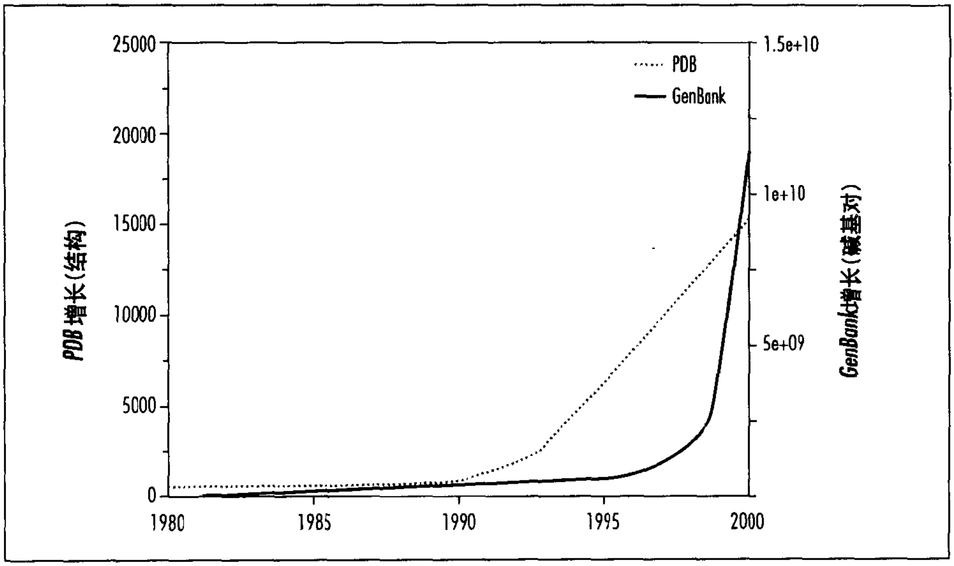


图 1-4　GenBank 和 PDB（Protein Data Bank）的增长极快

生物信息学的功能是表示、存储和发布数据。数据格式和数据库的设计，查询数据库工具的创建，开发用户界面把不同工具结合在一起，以使用户可以对数据进行复杂提问，都是生物信息学基础设施开发所涉及的领域。

开发从数据中发现知识的分析工具在生物信息学中是第二位的，也是更加科学化的方面。在许多水平上，我们可以应用生物学信息，通过比较序列来推测一个新发现的基因的功能；将已知的蛋白质的三维结构拆分成细小的部分，以推测蛋白质的折叠方式；或建立细胞中蛋白质和代谢物如何协同作用使细胞行使功能的模型。分析生物信息学家的终极目标是开发一种具有预测性的方法，使科学家可根据基因组序列来建立生物功能和表现型的模型。这是一个伟大的目标，只有通过许多科学家共同努力，才能逐步实现。

生物学给计算机科学家提出了哪些挑战？

在基因组计划时代，生物学的目标是量化地理解生物如何从编码它们的基因组形成自身。

破解基因组密码是非常复杂的。在最简单的层次上，通过基因序列的计算机分析标识未知的基因仍有许多困难。我们还不能对一个氨基酸链如何折叠成功能具备的蛋白质的特定结构进行预测和建模。

在单一的分子层次之外，也存在着巨大的挑战。在 GenBank 中，大量的数据正在呈指数增长，随着 DNA、RNA 之外的数据类型的出现，蛋白质序列之外的数据类型也同样扩增，仅仅管理、访问和以易懂的形式为用户显示这些数据也是一个很严峻的任务。为了管理这些大量的资料，人机交互专家需要和生物学方面的学术工作者以及临床研究者的工作紧密地结合在一起。

生物学数据非常复杂而且相互纠结。比如， DNA 阵列的一点，不仅和它的密度这个直接信息相联系，而且和多层面的信息如基因组定位、 DNA 序列、结构、功能和其他因素相联。创造一个信息系统使生物学家能够跟随这些链接，而不迷失在信息的海洋中，对计算机科学家来说也是一个巨大的机遇。

最后要说的是，基因组中的每一个基因都不是独立存在的。多种基因的相互作用形成的生物化学途径依次进入其他途径中。生物化学是受外界环境影响的，可以和病原体或其他刺激物相互作用。将基因组学和生物化学数据一起放到定量可预测的生物化学和生理的模型中，将是这一代计算生物学家的工作。计算机科学家、数学家和统计学家将成为这一领域的重要部分。

生物信息学家应该有什么样的技巧？

如果你有兴趣进行生物信息学研究，主题将是非常广泛的，但是一个人不可能全部掌握这些技巧。与工作在 Celera Genomics 和 Eli Lilly 公司的科学家谈话时，我们挑出了以下对生物信息学家的“核心要求”：

> 应该在分子生物学的某些方面具有深厚的基础，可以是生物化学、分子生理学、分子生物物理，或者分子建模等方面。但是如果没有分子生物学的核心知识，就像有人说的那样，会“经常碰壁”。
>
> 必须完全理解分子生物学的中心法则，理解DNA序列怎样和为何转录为 RNA 并翻译为蛋白质是至关重要的。（在第二章中我们将定义中心法则，同时回顾转录和翻译过程。）

应该有至少一种或两种主要的分子生物学软件包的实际应用经历，无论是在序列分析方面还是在分子建模方面均可。学会这些软件之一的经验可以帮助你更容易、更快速地学会其他软件。

应该习惯在命令行计算环境下工作。在 Linux 或 Unix 环境下工作可提供这种经验。

应该具有运用计算机语言如 C/C++以及脚本语言如 Perl 或 Python 的经验。

在这个背景下，可以加入一系列其他的先进技术部分：分子进化和系统论；物理化学 ——动力学、热力学和统计学结构；统计学和概率方法；数据库的设计和补充；算法的发展；分子生物学实验室方法，以及其他等等。

为什么生物学家应使用计算机？

对于理解任何可以以数学方法进行描述的系统来说，计算机都是一个强有力的工具。随着对生物学过程的认识的深入，生物学家应用计算机并不令人惊讶。计算生物学的学科以及近来的生物信息学是由经典的生物学、数学和计算机科学相互交叉而形成的。

数据收集的新方法

生物化学经常被认为是一种古老的科学。如果注意到一个有趣的疾病或者是有趣的特征，那么你很可能想了解这方面研究发展的情况。根据生物化学家对一特定的生物化学过程的兴趣，他可以同时确定一个简单的基因产物的序列、结构、分析其表达的性质。这样就会导致对于一条生化途径或一个蛋白质的详细的了解。然而，由于缺少进行这方面工作的方法，或者是要做一个特定的实验而不能有效地和其他科学家进行交流，这个途径或蛋白质与其他生物成分的相互作用经常不得而知。

因特网（Internet）已经改变了科学家分享资料的方式，并使得一个中心信息数据库服务于整个研究领域变成可能。但是，更重要的是，实验技术的快速发展，可以想像在一个中心“工厂”系统地收集特定类型的多种资料，并将它们分给多个研究者进行解释也是可能的。

20世纪90年代，生物学界开始实施一项史无前例的计划：对人类基因组的所有DNA进行测序。即使人类基因组序列的第一张草图已经完成，自动测序机仍然远未停止，以确定通常用于生物学研究的多种生命形式的基因组的全部序列。而且我们还要好好整理过去10年间收集到的人类基因组的数据。在这么多资料中，只有很少的相对比较重要的基因的定位是清楚的，而且这些资料还在增加。运用图像处理技术，整个基因组图像的产生比用化学的图像技术产生要快得多，但是即使应用这种技术，完成基因组数据完整和详细的图谱仍旧需要许多年。

近年来，X射线晶体学技术已经得到显著提高，它可以在几分钟内得到一个蛋白质完整的晶体反射，而不需要用几小时或几天。自动化分析软件可允许其在几天或几星期内完成结构确定，而不需要几个月。和人类基因组计划中序列确定方法一样的高通量的结构确定方法成为可能。虽然蛋白质的结晶仍然是限制速度的一步，在之后的5年至10年中，可用来研究的蛋白质结构的数目会越来越大。

并行计算这个概念存在好长一段时间了。将一个问题拆分成细小的可进行计算的分支部分，让处理器同时各自解决其中的一个小问题，而不是在一段时间内只解决一个问题。并行方法应用于实验分子生物学技术，如DNA微阵列。微阵列技术允许研究者在一个小的芯片上同时进行上千个基因表达的实验。微小的并行实验绝对需要可收集和分析数据的计算机的支持。他们也需要将数据进行电子发布，因为对于数据分析者来说可能是肤浅的数据信息也可能会让其他人感兴趣。通过查找这样的数据库找到的信息可以使科学家在实验室少工作几年时间。

这些实验工作的丰硕成果可以分享，这要归功于万维网的发展以及万维网促成的信息交流和交换的发展。

实验分子生物学自动化程度的提高以及信息技术在生物科学中的应用，导致生物学研究的方式发生了根本的改变。除了以前在一段时间对一个基因仔细地进行研究定位之外，我们现在可对所有可能得到的资料进行编目，完成完整的图像，然后再反过来对感兴趣的位点进行标记。这是在序列和结构的功能域中显示出来的情形，现在也逐渐应用于其他类型数据。其发展趋势是在公共的数据库中存储多种类型的原始的生物学资料，并且这种数据库对于进行科学研究的学术界是开放的。现在最初的研究工作不是在实验室中进行的。科学家可首先应用数据库查询来节省时间和资源。

要进行生物信息学研究应该怎样设置PC机？

迄今为止，你可能是通过使用 Windows 或 MacOS等用户界面良好的操作系统上运行的文字处理软件、而熟悉怎样应用PC机的。为了充分进行生物信息学的研究，需要学习Unix，对于功能强大的服务器或工作站来说，Unix是一种经典的计算机操作系统。绝大多数科学软件都是在Unix下开发的，真正的研究者肯定要用到许多只能在Unix系统下运行的程序。Unix有好几个版本，最流行的两个是BSD和SunOS（译注 1）。然而，最近有了第三种选择：Linux。Linux是一种开放源代码的 Unix操作系统。在第三、四、五章，我们将讨论怎样在 Linux 下建立工作站以运行生物信息学程序。其中包括操作系统及其如何工作：文件如何组织，程序如何运行，进程如何管理等，最重要的是，怎样键入命令使计算机快速执行你想进行的操作。

为什么要使用 Unix或Linux？

在我们的计算机上安装了Linux操作系统后，可以利用为Unix系统开发的最新的科学研究工具。随着在市场上变得越来越流行，Linux继承了Unix系统在开发、编辑和运行程序，联网以及多用户的管理等方面的强大功能，而且提供了桌面 PC的标准工具，包括文字处理工具、绘图工具甚至包括可视化编程工具。本书是以这样的假设为前提的：你愿意学习在 Unix系统上如何工作，而且你将在有Linux或将要安装其他Unix版本的机器上进行工作。对于我们要讨论的许多专门的生物信息学工具来说，Unix 是最实际的选择。

然而，在大多数Mac或PC环境中，Unix对于办公应用而言却不一定是最现实的选择。随着Linux操作系统的进一步普及，Linux上的文字处理、桌面出版软件以及外围设备会进一步改善。然而，它仍不能成为这些领域中消费操作系统的排头兵。只要知其所以然，使用Linux并不比使用普通的PC操作系统更难，但是起初需要的技巧以及可能遇到的问题都是陌生的。

译注 1：通称 Solaris。

注意：迄今为止，除了有几天硬件出故障外，我的台式机已经可靠运行Linux近5个月了。在这期间，没有出现软件冲突，也没有出现不友好的界面，而且也不会丢失 *.dll文件或奇怪的错误消息。当我第一次用它时，安装 Linux需要花费大约 2天的时间和一些来自技术支持的帮助，第二次安装时，只需要 1小时。实际上，我（在 Mac/PC环境中惟一的 Linux 用户）遇到的主要问题是如何打开来自 Mac 用户的邮件附件。——CJG

幸运的是，有些公司在销售封装好了的 Linux操作系统时，已使安装程序自动化了，而且对于安装提供90天的电话和 Web的技术支持。Red Hat和 SuSE这样的公司和Debian这样的组织为 PC机提供了 Linux 的发行版，而 Yellow Dog （或其他）公司为 Macintosh计算机提供了 Linux发行版。

有一系列方法可以使你逐渐适应Linux 。当然，如果有不止一个计算机工作站，你可以用一个电脑将其操作系统转变为 Linux ，而在其他电脑上保留你所熟悉的操作系统。另一个选择是安装两个操作系统。在这种情况下，可以在你的硬盘上建立两部分（叫做分区），在其中之一上建立 Linux 操作系统，而在另一个分区上保留原有的操作系统，因此，当你打开计算机时，可以有两种选择，一种是开始运行 Linux，另一种是运行其他的操作系统，你可以原地保存原来的老文件及程序，而在 Linux 部分进行新的工作。Linux的新版本，比如适于 PowerPC的 Yellow Dog Linux，允许用户在 Linux上仿真 MacOS环境，在两种平台上同时得到软件及文档。

我们能找到什么信息和软件？

在第六章我们将提到阅读信息的能力。只是在几年前，生物学家必须知道怎样应用纸介质的索引进行文献检索，然后再从相应的技术期刊上查找文献。现在，生物学家用基于 Web的数据库来得到同样的信息，并可得到数十倍的其他信息类型。无论是否进行计算，知道如何查找这些资源对每一个生物学家来说都是一种非常重要的技巧。

接下来我们将介绍在网上进行下列工作所需要的基本工具：定位数据库、计算机程序以及其他资源，并将这些资源转移到你自己的计算机上。一旦得到之后即可运行它们。从第七章到第十一章，我们将转到特殊类型的科学问题以及回答这些问题所需要的工具的问题上来。在某些情况下，有些计算机程序逐渐成为解决某一特定问题的标准（比如，用 BLAST 和 FASTA 来进行氨基酸和核苷酸序列的比对）。在其

他情况下，解决问题的方法可能还是一个开放的研究性的问题，可能有一些可与之竞争的工具，也可能根本没有可以完全解决问题的工具。

为什么需要从 Web 安装程序？

处理大量的复杂数据需要一个系统的、自动化的方法。如果需要在一个数据库中查找一个查询的匹配结果，可以通过 Web 表单解决问题。但是如果想查找与 10000 个查询相匹配的结果，还要对所得到的信息进行分类再发现这些结果的关系时怎么办呢？你肯定不想用一个 Web表单输入 10000 条查询，而且你可能也不需要为了在网页上看着漂亮一点而格式化返回的结果。共享的公共 Web 服务器速度经常很慢，用它们来处理大量的资料是不现实的，在第十二章将讲述如何利用 Perl 作为一个驱动器，帮助你使用自己的计算机，使用自己喜欢的程序来处理大量的数据。

不上课我就能学会一门编程语言吗？

任何人，只要有过为解答某个问题而设计并做过实验的经历，都具备计算机编程的基本技巧。实验室中进行的实验是始于一个问题，然后设计一个可检测的假想，也就是说，根据一个实验或数个实验的结果可以检测假设是否为真理。为检验假设而进行的实验过程和计算机编程是类似的。实验的本质是：如果你研究系统 X，并对它进行某些处理，将会发生什么现象？进行的实验必须仔细设计以使其结果能够解释清楚。计算机编程也必须仔细地设计，从而使一些值从一部分传递给另一部分时能够解释清楚。程序员必须建立一个明确的计算机指令，并且事先想清楚，不同类型的结果意味着什么以及计算机能处理什么。大部分有价值的计算机程序具有严密思考的能力，设计一个过程来回答一个问题，理解对于不含糊地回答问题需要的是什么。

即使具备这些技巧，学习一门计算机语言也不是一件轻而易举的事情，但是近年来随着 Perl 语言的逐渐发展，学习计算机语言也变得容易多了。 Perl语言被它的创造者认为是“因特网的输送管道，其他任何东西的输送管道”，开始时将它设计为处理数据的最佳脚本语言。但它逐渐成为一个全面的编程语言，使用 Perl 来开发任何真正的计算程序都是可实现的。Perl是一种非常灵活的语言，你可以只学习为解决一个独立的问题而编写脚本，在你进行了一次或两次这样的操作之后，就会拥有建立

它的一些核心知识。学会 Perl的关键是应用并且马上应用它，就像只读课本你不能流利地说出西班牙语一样，只读过 O'Reilly公司出版的《Learning Perl》（译注 2）也不能学会 Perl，设法用 Perl来尝试一下是有帮助的。第十二章我们将提供 Perl代码的例子，来解析普通的生物学数据类型的语法，以驱动其他编程语言程序的输出，甚至可用 Perl实现解决普通的计算生物学问题。我们以此来抛砖引玉，以激励你自己编写一些小程序。

怎样利用 Web 信息？

第六章将介绍可存储生物学数据库的公共数据库，它可被全世界的研究者共同分享。

使用 Web表单填写并搜索公共的数据库，快速找到一个蛋白质结构文件或DNA序列文件的同时，可能最终的结果是你想使用不止一个数据。你甚至正在收集和整理属于你自己的资料；你可能想为更广泛的生物学界提供新的数据，为了能够高效地进行这些工作，需要在自己的计算机上存储这些数据。如果要用计算机程序来处理存储的数据，需要组织自己的数据。理解组织的与未组织的数据的差异，并设计适合存储数据和符合需要的数据的格式是使数据有用和可访问的关键。

有许多种组织数据的方式。绝大多数生物学数据在平面文件数据中存储，当数据存储量很大时，这种类型的数据库会变得非常不方便。第十三章将介绍需要告诉数据库专家以及建立自己的数据库需要的基本的数据库的概念。我们将讨论普通的文件和相关的数据库的区别，介绍管理数据库最好的公共工具以及怎样应用它们来存储及获取所想要的数据。

怎样理解序列比对数据？

没有可视化工具的存在，很难讲清楚自己的资料或证明一个观点。为了讲清楚生物学数据，经常需要提取复杂的多变量数据的交叉部分。在结构数据库中存储自己的数据将在第十三章讨论，这是创建复杂数据分析的基础。

译注 2：本书中文版《Perl语言入门》已由中国电力出版社引进出版。详情请访问：*http://www.infopower.com.cn*。

在能够以一种可访问的、灵活的形式存储数据之后，下一步是提取重要的内容并使之可视化。无论是需要将数据做成直方图，还是以三维的形式显示其结构并观察它在实际情况下的运动，都有一些可视化工具可用。在第十四章中，我们将介绍数据分析和数据可视化工具，从一般的绘图工具包到与域有关的标记生物学序列比对的程序、显示分子结构、创建进化树以及可用于其他目的的程序。

怎样编写一个程序来比对两个生物学序列？

任何计算科学的重要部分都是知道什么时候需要自己编写程序，什么时候能够用其他已写好的代码。高效率的程序员是一个懒惰的程序员；如果已有很好的可用的程序，他绝对不会再去写它，如果你要找的是像两个蛋白质序列比对这样的常规程序，可以肯定别人已经写过了，查找一下就会找到一些源代码来参考。类似地，许多数学和统计学问题也可以用代码仓库中可得到的标准代码去解决。 Perl 程序员可在模块中得到简化操作的标准代码。有许多可免费获得的管理与 Web 相关的过程的模块，其中有创建为处理生物学序列数据标准模块的方案。

怎样通过序列来预测蛋白质结构？

有许多问题我们不能够回答，通过序列来预测蛋白质结构即是其中之一；实际上，它是计算生物学中最大的开放研究问题之一。我们所能给你的是发现这些问题的信息，以及其他人正在开发的解决问题的工具，甚至是给你激发灵感。生物信息学像其他科学一样，并不总是能快速而容易地解答问题。

生物信息学能回答什么问题？

现在驱动（和资助）生物信息学研究的问题，也是近几百年来在应用生物学中人们正在努力研究的问题。我们该如何治愈疾病？如何阻止传染病的发生？怎样才能产出足够的粮食养活全人类？从事开发药物、农用化学产品、杂交植物、塑料及其他无污染生物业务的公司，正在进行生物信息学开发，并寄希望生物信息学能提供新的目标，对替代稀有的自然资源有所帮助。

基因组计划的存在意味着我们可以利用他们提供的数据。简单来说，现代分子生物学的目标是解读所有生物的全部基因组。标识每个基因，用每个基因与其编码的蛋白质进行匹配，来确定每个蛋白质的结构和功能。我们希望基因序列、蛋白质结构和功能及基因表达模式的详细知识能够帮助我们理解生命活动最可能的方式。其中所蕴含的是人类拥有精细而准确地管理生物的能力。

第二章

生物学问题的计算方法

在生物信息学课程中讲授了一系列标准技术。目前大多数重要技术都是依据一个关键的原则：分子之间的序列和结构的同源性（homology，相似性）可用来表示结构和功能的相似性。在本章中，我们将对生物学家可应用的标准计算机技术进行概括。在本书的后面，我们将讨论专门的软件包怎样实现这些技术，以及如何应用它们。

分子生物学的中心法则

无论我们的研究进行得多么深入，理解细胞和分子生物学的基础都非常重要。如果你非常熟悉DNA和蛋白质结构、基因、转录及翻译的过程，可以跳过此节看下一节。

分子生物学的中心法则为：

> DNA作为模板可以进行自我复制，DNA可以转录为RNA，RNA可翻译为蛋白质。

由此可以看出，中心法则总结了基因组的信息功能。遗传信息是保守的，通过复制的过程传给子代。遗传信息可通过转录和翻译表现在个体中。许多层次比如在结构、生物化学和细胞层次的功能是在基因组信息的基础上形成的。但是，最终所有生命的功能都会回到基因组的信息内容上。

换一种说法，即基因组 DNA 包含生物主要功能的内容。如果没有 DNA，机体将不能复制自身。原始的“一维的”DNA序列，并没有任何实际的生物学功能；它只是信息，是一个蓝图，可以由细胞的蛋白质合成机解读。DNA序列是穿孔卡片；细胞是计算机。

DNA 是由称为核苷酸（nucleotide）或碱基（base）的单个化学单位组成的线性多聚体。组成（至少是在地球上存在的）生物 DNA 序列的四种核苷酸分别为腺嘌呤、鸟嘌呤、胞嘧啶和胸腺嘧啶——分别用 A、G、C 和 T 来表示。在线性 DNA 序列中核苷酸的顺序包含构建有机体的指令。这些指令在复制、转录和翻译过程中被解读。

DNA 的复制

DNA 分子结构的特殊性赋予了 DNA 特殊的性质。这些特性允许存储在 DNA 中的信息得以保留，并且从一个细胞传递给另一个细胞，而且可以从父代传向子代。DNA 的两个分子组成一个双螺旋结构，其全长（可能由几百万个核苷酸组成）以一定的规则进行配对。在每条 DNA 链中，双螺旋结构的每条链通过核苷酸的氢键相互结合在一起。核苷酸以特殊的方式进行键合。A 和 T 配对，G 和 C 配对，每一个配对称为碱基对（base pair），DNA 序列的长度由碱基对（或 bp）、千碱基对（Kbp）、百万碱基对（Mbp）来表示。

DNA 双螺旋的一条链是另一条链的“镜像”结构。在一条链中为 A，在另一条链中相对应的就为 T。如果一条链中为 C，则另一条链中通常为 G。

当一个细胞分裂形成两个新的子细胞（daughter cell）时，双螺旋 DNA 通过解螺旋后进行复制，每一条链都可以作为模板来合成它的化学镜像，或称互补（complementary）链。这个过程如图 2-1 所示。

基因组和基因

编码生物的整个 DNA 序列称为基因组（genome），然而，基因组不是通过一个序列的全长来发挥作用的。基因组被划分为独立的基因。基因（gene）是一个小的、整个基因组序列定义的部分，每一个基因都有一个特定及惟一的作用。

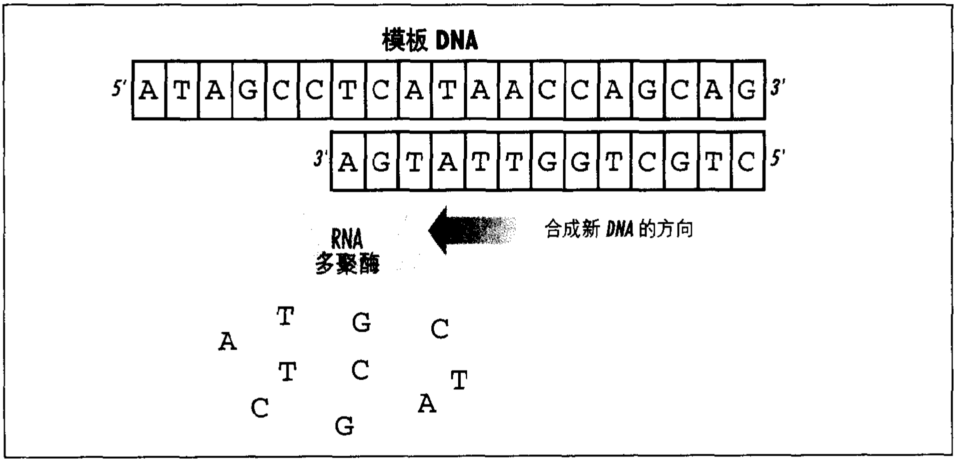


图 2-1 DNA 双螺旋中的一条被复制的示意图

一共有三种类型的基因。编码蛋白质（protein-coding）的基因是合成蛋白质分子的模板。每个被基因组编码的蛋白质（protein）在机体中都是一个具有特定目的的化学机器，RNA 有关的基因也是化学机器的模板，但是与组成蛋白质的模块不同。最后，非转录基因（untranscribed gene）是具有一些功能目的的基因组 DNA 区域，但是它们的目的不是通过转录或翻译创建其他的分子来达到的。

DNA 的转录

DNA 不仅可以是自我复制的模板，而且可以是 RNA 分子的蓝本。DNA 转录为 RNA 的过程称为转录（transcription），如图 2-2 所示。RNA 在结构上和 DNA 相似。它可以是由单个化合物单元组成的多聚分子，RNA 链和 DNA 链的化学骨架有所不同，允许 RNA 以一个单链的形式存在，或以双螺旋的形式存在。单链分子在 RNA 链的不同部分形成碱基对，使 RNA 折叠成三维结构。RNA 的单个化学单元为 A、C、G 和 U（尿嘧啶，代替胸腺嘧啶）。

基因组为一系列 RNA 分子提供模板：RNA 分子的三个主要类型为：信使 RNA、转运 RNA 和核糖体 RNA。信使（messenger）RNA（mRNA）分子是基因转录的产物 RNA。它们将信息从基因组转运到核糖体这个细胞蛋白质合成的装置。转运（transfer）RNA（tRNA）分子是转运组成蛋白质的氨基酸分子至核糖体的非翻译 RNA 分子。最后，核糖体（ribosomal）RNA（rRNA）分子是核糖体的非翻译 RNA

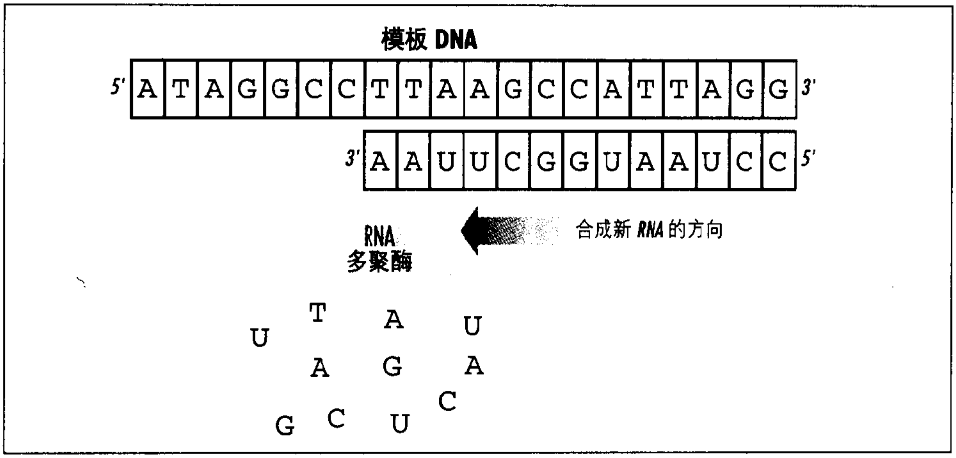


图 2-2　DNA 转录为 RNA 的示意图

成分，核糖体为蛋白质和 RNA 的复合物。rRNA 参与锚定 mRNA 分子，并催化翻译过程的一些步骤。一些病毒也用 RNA 来代替 DNA 作为遗传物质。

mRNA 的翻译

将 mRNA 翻译为蛋白质是将信息从基因组最终转移至活细胞的主要步骤。

像 DNA 一样，蛋白质是一个线性的多聚体，由一个个代表化学变量单元的字母组成。蛋白质的字母是一系列称为氨基酸（amino acid）的小分子。

和 DNA 分子不同，蛋白质的化学序列有其信息内容也有物理化学的“内容”。一般组成蛋白质分子的 20种氨基酸中的每一种都有不同的化学性质，其性质是由它的侧链（因氨基酸的不同而异的化学集团）来决定的。蛋白质的化学序列称为蛋白质的一级结构（primary structure），对于蛋白质的功能来说，序列折叠形成一个紧凑的分子的方式和蛋白质的一级结构一样重要。蛋白质的二级结构和三级结构使蛋白质完成最后的折叠，可以使蛋白质的化学序列相隔较远的部分凑在一起形成功能位点。

如图 2-3 所示，遗传密码(genetic code)是将 DNA 翻译为蛋白质的密码。它由 DNA 的三个代码（称为密码子，codon）来编码蛋白质序列中的每个氨基酸。简单的组合显示在 4种核苷酸中选择 3种氨基酸可组合成 64 种方式。事实上，存在 64 种可能

的密码子，而只有 20 种氨基酸。说明有些密码子是过剩的。而另外的密码子具有特殊的功能，提示细胞的翻译机器停止翻译 mRNA 分子的过程。图 2-4 显示 RNA 怎样翻译为蛋白质。

	第二个位置								
第一个位置	U		C		A		G		第三个位置
U	UUU	Phe	UCU	Ser	UAU	Tyr	UGU	Cys	U
	UUC		UCC		UAC		UGC		C
	UUA	Leu	UCA		UAA	Stop	UGA	Stop	A
	UUG		UCG		UAG	Stop	UGG	Trp	G
C	CUU	Leu	CCU	Pro	CAU	His	CGU	Arg	U
	CUC		CCC		CAC		CGC		C
	CUA		CCA		CAA	Gln	CGA		A
	CUG		CCG		CAG		CGG		G
A	AUU	Ile	ACU	Thr	AAU	Asn	AGU	Ser	U
	AUC		ACC		AAC		AGC		C
	AUA		ACA		AAA	Lys	AGA	Arg	A
	AUG	Met (start)	ACG		AAG		AGG		G
G	GUU	Val	GCU	Ala	GAU	Asp	GGU	Gly	U
	GUC		GCC		GAC		GGC		C
	GUA		GCA		GAA	Glu	GGA		A
	GUG		GCG		GAG		GGG		G

图 2-3　遗传密码

分子进化

DNA 复制和翻译中发生错误是很常见的。如果这些错误发生在有机体的生殖细胞，它将传递至子代。DNA 序列的改变称为突变（mutation)。突变可以产生有害的结果，子代很少能活到成年，突变也能产生有益的结果，或者是无意义的结果。如果一个突变不能在个体出生之前将它杀死，这个突变将在种群许多世代中保持稳定。这些改变缓慢的积聚导致了进化（evolution）的过程。访问 DNA序列可使人们对进化有更深一步的了解。人们将进化的分子机制理解为 DNA序列突变积聚的过程，正好成为以 DNA和蛋白质序列比较为基础的正在发展的假说的一个证据。

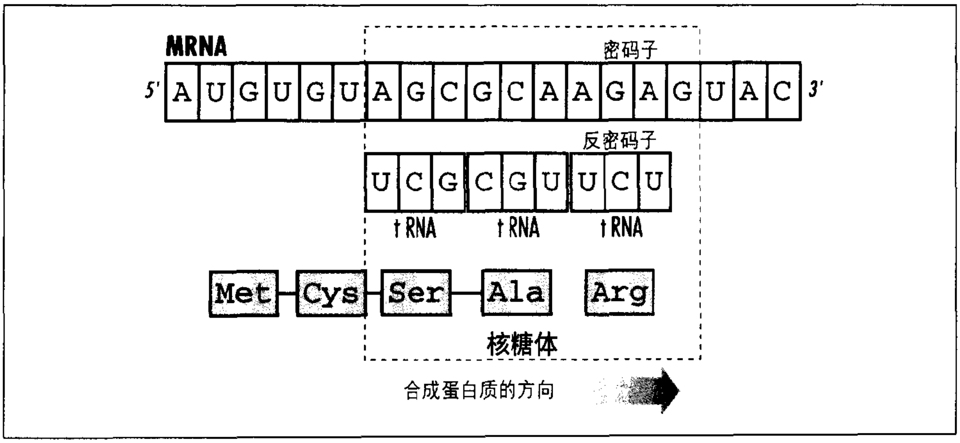


图 2-4　标准碱基对合成为蛋白质的示意图

生物学家要建什么样的模型？

上文我们对分子生物学进行了简短的回顾，下面看一下如何在分子生物学问题应用它。生物学和生物信息学的重要内容之一是建模（modeling）。模型（model）是指以简要的方式描述复杂的系统。将一个复杂的事物，如染色体、细胞分化周期简化为一个具有你想研究的所有特性的模型非常困难。模型可帮助我们看到更多东西。好模型的特性是可以用定量方法使对复杂系统的研究变为简单的分析。应用生物信息学工具从生物系统中提取相关的参数（此生物系统可以是单分子，也可以是像细胞一样复杂的系统），定量地描述它们，然后开发一些计算方法，应用这些计算方法可用这些参数来计算系统的特性或预测它的行为。

为帮助大家理解什么是模型，以及一个好模型可以做什么样的分析，可以参看下面三个基于生物信息学的例子。

通过分子的一维表示访问三维分子

事实上，DNA和蛋白质是复杂的三维分子，由成千上万甚至上百万个原子结合在一起。然而，DNA 和蛋白质都是多聚体（polymer），是由具有共同骨架的重复的化学单元（单体，*monomer*）链接在一起。多聚体的每个单体中都有两种原子，一种原子在每个单体中都不改变，组成多体的骨架，另一种原子在不同的单体中也不同。

在 DNA 中，四种核苷酸（A、T、G、C）一般用于构建多核苷酸链。在蛋白质中，20 种氨基酸用于构建多肽链。在 DNA 链中，四种核苷酸可以以任何顺序排列，其排列顺序决定此 DNA 到底是哪种 DNA。同样，在蛋白质中，氨基酸可以按任何顺序排列。蛋白质的顺序决定蛋白质的折叠和功能。

在弄清楚DNA和蛋白质的化学性质之后不久，研究人员就发现用单个字母来代表链的结构是很方便的。DNA 序列中的每个核苷酸不必用详细的化学结构来表示，可用 A、T、C、G 来表示。因此，一个包含上千个原子的短 DNA链，可以用只有几百个字母的序列来表示。图 2-5 中表示了一个多核苷酸链的简化方式。

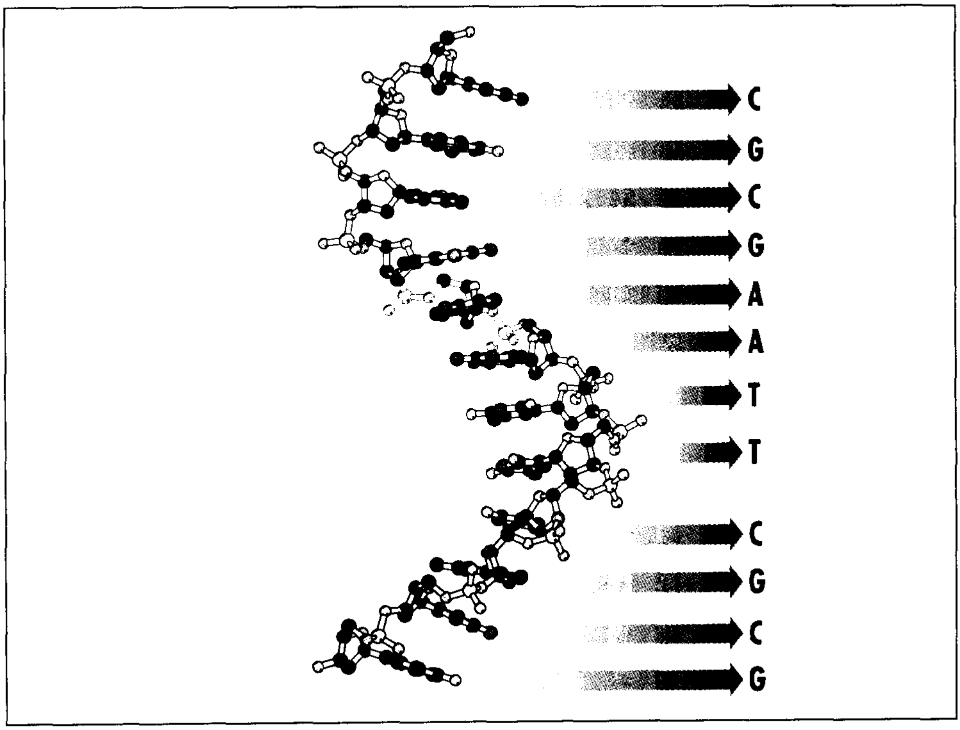


图 2-5　简化的多核苷酸链表示

这种简化的方式不仅可以节约存储空间，而且提供了共享序列信息较为方便的形式。它代表了分子独特的和准确的性质，并忽略了在实验时无法访问的细节（比如 DNA 和许多蛋白质的原子结构），许多计算生物学方法都利用生物学大分子的一维结构来提取其三维结构的性质。

将蛋白质和核苷酸序列简化为一维的链式结构，已经成为计算分子生物学中最有成效的建模策略之一，而对字符串的分析，是计算科学一个历史悠久的研究领域（注 1）。关于字符串一个基本的问题是匹配。在计算机科学中，已经有很好的算法可以寻找各字符串之间精确和不精确的匹配。这些算法可应用于生物学序列的配对，以及输入序列后对数据库的寻找。

除了进行单个序列的匹配外，计算机科学中以字符串为基础的方法已经成功应用于分子生物学中的许多其他问题。比如，使用从一系列短的子字符串重新构建的算法可以从相互重叠的序列片段装配DNA序列。从一个简单的序列中识别重复模式，或从多个序列中识别保守模式的技术允许研究人员鉴别与生物学结构或功能相关的形式。最后，多序列比对技术允许同时比较几个分子序列，通过比对，可能能够推断出这种序列间进化上的相互关系。

对DNA和蛋白质序列简化的抽象忽略了许多生物学细节。生物学分子的细胞环境完全被忽略，生物学分子与其他分子的相互作用以及它们的分子结构也被忽略。但是就像以前就一再说明的是，生物学序列之间的匹配是具有生物学意义的 —— 比如，发现人类和果蝇眼睛形成的基因的相似性（在第一章中讨论过）就是一个例证。

为蛋白质结构建模的抽象

生物学不仅仅是序列问题。蛋白质及核苷酸具有复杂的三维结构，提供了生物机体功能的线索。分子结构通常代表原子的集合，每一个原子在三维空间中都有一个确定的位置。结构分析可以在静态结构、运动和分子之间相互作用等方面进行，可以用分子模拟的方法来分析。

标准的分子模拟的方法是把蛋白质作为由化学键连接的质点（原子）的集合来建立蛋白质模型。两原子之间的键具有标准的长度值，此值来源于化学实验，在此长度时，键可使原子之间的键产生紧密结合的应力。三个毗邻原子的角度有一个标准值，围绕着此值的键角具有一个应力。同样，由四个毗邻的原子组成的二面角的情况也

注 1： 字符串（string）是指字符的连续序列。字符（character）是从已定义的字母集中选择出来的一个字母；可以是二进制码（0 和 1 的字符串）或更复杂的可以在计算机键盘上输入的字母和数字字母。

是一样。在分子代谢的模拟中，通过模拟“加热”可以将能量加入到分子系统。根据标准的牛顿流体法则，在分子中的各个原子是运动的。将能量加入到系统内，可以提供使分子中的原子脱离标准的构造的力量。在一个分子系统中，上百个分子的作用及反作用可以用这种抽象来模拟。

然而，分子模拟对计算的要求是很高的，而且在力场（force field，建立分子模型的标准力的集合）和非键合的相互作用（nonbonded interaction）建模中（非相邻原子的相互作用），都有一些不确定的因素。现在用由所有原子建立模型的方法来预测蛋白质的结构还不太可能。

有些研究人员在对简单的蛋白质拓扑学的预测方面已取得一些进展，他们运用了中等程度的抽象——比线性的序列要复杂，而比所有的分子模型简单。在这种情况下，蛋白质作为一条链（代表蛋白质的骨架）上一系列的小珠。小珠的不同代表了氨基酸侧链的不同，具有不同的性质。它们可带正电荷，或负电荷，有极性或无极性，体积可大可小。按照规则，其中的小珠之间相互吸引。同性相斥，异性相吸。极性集团与极性集团聚在一起，非极性集团与非极性集团聚在一起。在链中也有一些规则，主要是在模拟的过程中链不能穿过它自身。折叠过程的自身模拟，是通过每个位置的小珠相继的或同时的运动来完成的。

生化系统的数学建模

生物学理论模型的应用远远超越了单分子的水平。许多年来，生态学家不断应用数学模型来帮助人们理解相互依赖的群体代谢的变化。一个早期存在的物种的数量减少会对食物链中捕食的动物产生什么样的结果？对环境中的种群变化会产生什么作用？对这些问题的回答，是给定一个恰当的数学模型、群体规模大小的知识，以及根据多种因子变化的标准速率进行的理论上的预测。

在分子生物学中，一个与称为代谢控制分析（metabolic control analysis）相似的方法可应用于有许多分子和化学元素参与的生化反应。当细胞包含成百上千种相互作用的蛋白质、小分子和离子时，可以创造一个模型来描述与预测复杂代谢的一部分。比如，如果对于真核细胞中线粒体内膜两侧保持不同的氢离子浓度感兴趣的话，很可能在模型中不需要包括参与血红素结构的生物合成代谢途径的远端部分。

代谢模型描述了与一个途径的化学物质浓度相关的生物化学过程，以及影响它们浓度的反应和流量，反应和流量在不同的等式中描述，它们是重要的浓度变化率。使代谢模拟有意思的是为许多同时进行的反应建立模型，以观察它们对于特定化学物质的浓度产生作用的可能性。应用一个构建好的代谢模型，可以检测细胞状态的不同假说，以及微调模型来拟合实验结果。结果，它们可以提示可测试的假说来推动进一步的研究。

为什么生物学家要建立模型？

我们反复提到，理论建模提供了可检测的假说，但不是结论性的答案。但是，有时做出这种区分是不太容易的。尤其是成对的序列比较时，好像是已经提供了答案。根据序列相似性鉴定基因最终还需要实验的证明。根据一个未知的DNA序列和已知其详细性质的基因序列相似，而认为它们具有相似的功能，这样的结论，其根据是不充分的。两个序列可能亲缘关系较远，由于进化的原因已具有不同的功能。然而，有理由将序列的相似性作为证实其功能相似性的起始点；如果序列的同源性提示一个未知基因和柠檬酸合成酶相似，那么首先应该进行的实验是检测未知基因产物的柠檬酸合成酶活性。

在生物学研究中，应用计算工具的一个主要好处是，这时选定分子生物学和生物化学实验的目标就更容易了。应用从序列简图（profile）方法到蛋白质结构的几何和物理化学分析中的任何方法，研究人员都只能集中关注可能在功能上比较重要的狭窄的序列或结构部分。仅仅在10年前，这种关注还只能用位点受控突变（site-directed mutagenesis）的“鸟枪(shotgun)”方法来进行，此方法必须产生蛋白质的随机单残基突变，并描述其特征以选择可能的目标。功能基因组学和代谢重建方面的研究已开始为生化学家进行深入研究提供一个框架。

对于正在关注生物信息学研究方法发展的研究人员来说，对数据的一般规律及性质的发现是计算机记录的最感兴趣的问题类型。它也有多种类型，而且还不能给出许多规则。研究者已经从序列的模式到分子结构的原子的分类中发现了有趣和有用的性质，并应用这些性质发现并建立了genefinder、二级结构预测工具、简图方法以及同源性建模工具。

生物信息学研究者正在致力于现在成功率可能很高的问题的研究，比如从 basecalling 到序列比对、蛋白质结构建模以及基因组的比较，试图提高其精确性以及加大这些过程的范围。信息技术专家正在建立数据库结构和查询工具，从基因表达数据到分子内的相互作用。像其他领域的研究一样，生物信息学的研究也有许多可获得灵感的研究领域，找到它们的惟一方式是深入研究现在的文献。

本书覆盖的计算方法

分子生物学研究是一个快速发展的领域。可收集到的数据数量与数据类型正在快速膨胀，在公共数据库中存储这些数据的趋势已经从基因组序列这种数据类型扩展到其他生物学数据类型。对生物学家来说，信息变化是如此之快，以至于我们在本书中阐述的内容在面世之前就可能有点落后。

然而，从人类基因组计划开始以来，已经出现了一组核心计算方法的核心部分，可以处理在公共数据库（ DNA蛋白质序列、蛋白质结构）中可以共享的数据的类型。尽管包含高通量的分子生物学方法的实验结果数据库还没有序列数据库那么庞大，但是已经出现了分析这些数据的标准方法。

下面将简单地介绍本书中提到的计算方法：

使用公共数据库和数据格式

对于生物学家来说，首要的关键技巧是学会使用在线搜索工具来查找信息。文献检索已经不再是从印刷的索引中来查找文献了，可以在线查到所需的大多数科学出版物。网上有收集文献信息的中心数据库，因此可以马上查找到大量的期刊。你甚至可以建立“代理”来关注你感兴趣的领域中出版的新文章。搜索公共的分子生物学数据库需要的技巧和文献检索一样：需要知道怎样建立一个可以从数据库中查找的特殊搜索工具。查找生物化学文献和序列数据库的工具将在第六章中介绍。

序列比对和序列查找

在第一章中曾经提到过，能够对 DNA 或蛋白质序列进行成对比较并提取出部分的匹配，使得运用生物学序列作为数据库查询成为可能。对于生物学家来说，以序列为基础的查找是另外一个关键的技巧。在开始进行一个课题研究时，利用生物学数据库可以节省出许多时间。同源序列的鉴定提供了系统发育

分析及序列模式识别的基础。可以通过Web的形式在线进行以序列为基础的查找，因此它不需要特别的计算机技巧，但是为了判断查找结果的质量，需要弄明白序列比对方法是怎样运行的，除了简单的序列比对外还有其他类型的分析。成对序列比对以及在序列基础上的数据库搜索将在第七章中介绍。

基因预测

基因预测是试图检测未定性的DNA 序列中有意义的信号的许多方法之一。迄今为止，在 GenBank中存储的序列存放时已经被描述过。也就是说，已经有人运用分子生物学的、遗传学的以及生物化学的方法指出基因的功能。然而，既然基因组计划也正在全面开展，就会有许多 DNA 序列的性质需要描述。

用于预测开放阅读框、基因、外显子拼接位点、重复序列、 tRNA基因的软件可以帮助分子生物学家理解未标明的 DNA。基因预测工具将在第七章介绍。

多序列比对

多序列比对方法将许多相关序列的成对序列比对结果装配到一个基因家族所有成员的序列同源性图谱中。对于DNA或蛋白质序列功能较重要的位点的可视化鉴定，多序列比对可以有所帮助。这些位点一般比较保守。也就是说，在一系列相关的序列中，每一条序列的某一位点上都是同一个氨基酸。多序列比对也可用于定量分析来提取一个基因家族的信息。多序列比对是相关序列家族系统发育分析中必要的步骤，也能够提供可描述特定蛋白质家族的识别序列模式的基础。建立和编辑多序列比对的工具在第八章中介绍。

系统发育的分析

系统发育的分析试图描述一系列序列的进化关系。经典的系统进化树分类绘成图表可代表它们相对的进化多样性。远离根部的树的分支将单个的类别分开；靠近根部的分为界、门、纲、科、属等等。

分子序列比对信息可用来计算一个特定基因序列家族的系统进化树。根据序列相似性计分，或者是将一个序列突变为另一个序列需要的步骤数计算的信息理论模型，系统进化树代表进化上的远近。尽管随着基因组计划的实施可提供大量可用的数据以创建广泛的进化模型的能力大大提高，蛋白质序列家族的系统进化分析说明的不是整个机体的进化，而是在特定的编码区进化上的变化，系统进化分析的工具将在第八章中介绍。

从序列数据中提取模式及简图

基序(motif)是定义蛋白质亚结构的氨基酸序列，它们连接起来具有功能或者

结构上的稳定性。在与进化相关的基因序列组中，基序以保守的位点出现。在基因序列中趋于保守的位点（在一个序列族的所有或绝大多数代表中保持相同），在那些位点有突变的基因复制时，具有选择的压力。基因序列中不重要的部分在进化过程中可以变化，因此，保守的基序区域是突变噪声之中的一个信号。序列的简图(profile）是对于这些基序信号的统计学描述。简图分析可以通过挑出甚至与同一家族中其他成员根本不同的基序信号，帮助识别进化上关系较远的蛋白质。分析简图及发现基序的工具将在第八章中介绍。

蛋白质序列分析

蛋白质序列的氨基酸组成是许多分析的基础，从计算蛋白质的等电离点和分子量、用特定的蛋白质酶消化后形成的特征性肽质量指纹（peptide mass fingerprint） 到预测二级结构性质和翻译后修饰位点。性质预测的工具将在第九章中介绍，进行蛋白质组学分析的工具将在第十一章中介绍。

蛋白质结构预测

通过实验确定一个蛋白质的结构，比获得 DNA 序列的数据要困难得多，生物信息学及计算生物学研究中一个非常活跃的领域，就是根据蛋白质序列来预测蛋白质结构方法的开发。二级结构预测和穿线法 （threading)有助于确定蛋白质怎样折叠，与具有同样拓扑结构的其他蛋白质分类，但是不能提供详细的蛋白质结构模型。最有效及实际的蛋白质结构预测的方法是同源性建模（homology modeling)，即运用一个已知结构作为模板来建立一个相似序列的结构模型。如果没有同源性，则没有办法可预测一个蛋白质的完整的三维结构。蛋白质结构预测的工具将在第九章中介绍。

蛋白质结构性质分析

蛋白质结构具有许多可测量的性质，引起了晶体学家及结构生物学家的兴趣。蛋白质结构验证工具被晶体学家所用，他们用它来测定一个结构模型与从现存的结构或化学模型化合物中提取的结构规则的相符程度如何。这些工具也可分析在一个结构模型中每个氨基酸的环境“适合度”，以免在没有相反电荷的情况下，电荷的包埋以及在蛋白质表面发现疏水性氨基酸这样的怪事出现。这些工具对于评估实验性以及理论性结构模型非常有用。

另一类工具可以计算蛋白质的内部几何以及物理化学的性质。这些工具通常用于帮助开发蛋白质催化机制或其他化学性质的模型。蛋白质结构中最有意义的性质之一，是深的凹表面裂缝及内窝的定位，二者都可能指向辅助因子结合位

点或活性位点的定位。其他工具可以计算氢键模式或分析分子内的相互作用。一个特别有趣的性质是包围蛋白质与其他静电控制参数的静电场。这些参数包括单个氨基酸 pKas，蛋白质溶解能及结合常数等等。蛋白质性质分析的方法将在第十章中介绍。

蛋白质结构比对和比较

即使两个基因序列没有明显的同源性，它们编码的蛋白质的结构也可能是相似的。计算结构相似性的新工具，可通过对比结构，甚至是序列很不相似的结构检测距离较远的同源关系。这些工具对于根据已知的蛋白质结构比较结构同源性模型非常有用，蛋白质结构比对工具将在第十章中介绍。

生物化学模拟

生物化学模拟应用动力学系统建模对参与代谢的生物化学反应进行模拟。可以模拟单个代谢途径，也可以模拟跨膜运输的过程，甚至整个细胞或组织的特性。传统的生物化学及细胞模拟依赖于科学家用数学方法描述一个系统的能力，即开发可代表系统中不同反应以及发生变化的不同等式。然而，新的软件工具可以根据使用者提供的描述，建立模拟的数学框架。动力学使得对于任何对系统有足够了解，可以按照动力学建模惯例来描述系统的生物学家都可以进行数学建模。动力学系统建模工具将在第十一章中介绍。

整个基因组分析

随着越来越多的基因组序列的完全预测，原始的基因组数据分析成为一个非常重要的任务。从基因组数据中可以得出许多内容，比如，基因组数据可以看作是一个长的线性序列，但是用现存的遗传学以及物理学的图谱资料来整合 DNA 序列信息非常有用。它允许从一个非常大的基因组中进行搜索，并找到你想要的东西。NCBI 和其他组织正在努力提供对基因组数据有用的 Web 界面，使用户能够从一个高水平的图谱搜索一个特定基因序列的定位。

然而，基因组搜索不仅仅是基因组序列的分析。基因组的注释框架（它将基因查找分析结果以及序列同源性信息与基因组序列整合起来）正在变得越来越常见，而且已经开始面对基因组之间完全成对比较的进行与分析上的挑战。基因组分析工具将在第十一章中讨论。

引物设计

许多分子生物学方法中需要设计寡核苷酸引物（primer)。恰当的引物设计对于 PCR（polymerase chain reaction)、寡杂交、DNA 测序、微阵列实验的成功

是非常关键的。设计的引物必须与靶 DNA 杂交，从而对于提出的问题给出一个明确的答案，但是它们也必须有恰当的物理化学性质：不能自身杂交或者二聚化，在进行研究的序列中不能有多个靶位置。有几个基于 Web 的服务允许用户输入 DNA 序列并自动检测 恰当的引物，或者计算所需要的引物 DNA 序列的性质。引物设计工具将在第十一章中介绍。

DNA 微阵列分析

DNA 微阵列分析是一个相对较新的分子生物学方法，通过经典的探针杂交方法延伸而来，可以一次提供上千个基因的检测。由于微阵列实验结果的均匀、标准化性质（一个大小相等的点的栅极，每一个可辨认的栅极都有一个特定的 DNA 序列），其结果可以进行计算分析，对于大的微阵列分析需要计算工具，因为结果的图像看起来很复杂，以至于用手工进行比较非常不容易。

目前微阵列分析的主要任务是图像分析阶段（需要辨识在阵列图像中的单个点，量化信号强度）和聚集阶段（辨识相似信号强度的点）。在微阵列实验的芯片设计阶段，也需要计算的支持，为一组特定基因确定适合的寡核苷酸探针，以及在可能包含上千个的单个实验的栅极中保存每个点的识别记录。阵列分析工具将在第十一章中讨论。

蛋白质组学分析

在对蛋白质进行结晶及描述生物化学性质之前，经常使用凝胶电泳、部分测序及质谱结合来研究蛋白质。二维凝胶电泳可以将上千种蛋白质的混合物分为不同的组分，物质的单个点可以留下印迹，甚至凝胶上先切割然后进行分析。简单的计算工具在帮助分析蛋白质混合物的过程中会提供一些信息。根据蛋白质序列计算分子量及 pI 值是很容易的。将这些值结合起来，一系列特定的蛋白质可以在凝胶上的每个点中找到。当蛋白质通过具有特定的裂解位点的酶作用分解成为片段时，可以产生肽指纹。可以根据蛋白质序列来计算肽指纹。蛋白质片段的质谱分析可以比较计算出的肽指纹，以进一步限制搜索。蛋白质组学的工具将在第十一章中介绍。

一个计算生物学的实验

以计算机为基础的研究方案和实验数据的计算分析，必须遵循与其他科学研究同样的原则。研究结果必须清楚地回答开始准备测试的问题，如果其他人使用同样的输入数据及同样的过程，其结果必须具有可重复性。

如果你已经进行过实验生物学的研究，那么对于科学方法就会有一个很好的理解。尽管实验数据、方法及结果都编码存储在计算机文件中而不是放在实验台上，设计一个计算实验和你以前所做的设计一样。

虽然在这个自动化的时代，简单地把查询提交给搜索引擎并使用得到的结果不必对该过程有太多了解，这是件很容易的事情。但是我们需要用做一般实验室实验时同样的方式来理解方法并分析结果。有时候这些说起来容易做起来难。因此，让我们来看一下计算生物学中定义一个实验所包含的步骤。

问题的确立

科学实验总是始于问题。问题往往很广泛，比如，“蛋白质 X 的催化机制是什么？”通常来说，不可能总是通过一个实验来回答一个复杂的运行机制问题。这个问题需要分成若干部分，每一部分可形成一个假说。

假说（hypothesis）是可以通过实验来检测的陈述。在解决一个问题的过程中，可能会形成许多可检测的陈述，其中有的简单，有的复杂。比如，回答“蛋白质 X 的催化机制是什么”这个问题的第一步，可能提出这样一个假说：“在蛋白质 X 中的一些氨基酸，与在蛋白质 X 中具有相同功能的蛋白质是保守的”。通过计算机程序，你可以对许多与蛋白质 X 同类的蛋白质序列进行比对，然后寻找所有或大多数序列中相同的氨基酸。随后，你可能跳到下一个假说“在蛋白质 X 家族中有些保守的氨基酸与催化机制有关”。这个相当复杂的假说可以分解为一系列更小的假说，使每一个假说都可以进行检验（或者通过实验检验，或者通过另一个计算程序来检验）。

如果目标找的不对或者难以解答，那么研究课题就会变得没完没了。另一方面，如果不仔细的话，很容易在新信息的基础上在课题中添加新问题，使得当其接近完成时，总是不能达到目标。计算的课题做起来很容易，因为一旦开始时购买计算机及安装软件后，材料和资源的花费很少，看来在计算机上反复进行计算对于资源的浪费不大。

如果在进行某项计算课题之前，花一些时间写出课题的目标及时间安排的计划，就可以避免上述不正常状态的发生。如果要写一个课题计划书，从如下问题的答案写起来会很容易：

这个课题想回答一个什么问题？

你期待结果的最终形式是什么？目标是编写出计算机程序，给出一系列可用于正在研究的课题的数据还是在杂志上发表文章？需要什么才能成功或者完成课题 ？

课题大致的时间安排怎样 ？

课题一般的大纲是怎样的？这里应该将课题分为若干个组成部分，并描述完成每一部分需要做的工作。

你的课题为其他人安排的工作是否合适？如果你是独自一个人，你可以不必担心这一点，但是科学家进行研究是需要相互合作的。最好弄清楚你哪一部分的工作需要依赖别人。如果你正在写的是一组人共同研究的课题的计划书，应该指明每部分的负责人是谁。

从哪些方面来说再继续下去没有意义？

仔细想清楚这样类似的问题，不仅可对课题要达到的目标有一个清楚的思路，而且可列出用来组织研究工作的大纲。

将问题分解为更简单的部分

从第七章到第十四章，本书将涵盖在研究中使用生物信息学工具和数据库的一般过程。这些协议并不是尖端科学。开发自己的生物信息学研究的计算机技巧的关键，是知道什么工具可用，以及怎样使用。然后对你想解决的问题给出相应的办法，将它们分为不同的模块，如序列查找、简图检测、同源性建模、模型评估等等，对每一部分都要建立计算方法。

评价需要

将问题分解为模块时，应该根据可获取的数据和建模的起点，以及你需要什么，来评估一下你拥有什么。能够从 A 点到 B 点，从 C 点到 D点，不能帮助你确定是否绝对没有路线可以从 B点到 C点。比如，如果不能找出一个未知 DNA 序列的同源序列，那么不可能超越这一步骤而进一步去建立模型。即使你确实找到一系列具有确定简图的蛋白质，如果在 PDB 中没有同源结构的话，也不能根据你的研究计划而开

发一个结构模型。这只是从一般意义上来说，但是确实有些方式可以在投入时间和精力进行研究前得到你想要的结果。

选择合适的数据集

在实验室装备中，在实验中用到的材料是物理对象或物质。有必要记录有关材料的数据，如生产日期、由什么人准备，可能还要写明怎样准备等等。

在计算生物学中，也需要同样类型的文档，但差异是用数据进行，而不是对特定的物体或物质进行。你进行处理的数据资源应该区别于组成你实验结果的数据。可能会从许多生物分子数据库之一得到源数据。在第十三章中，我们可以学到更多怎样在数据库中存储信息，以及怎样提取出信息的知识。应该记录源数据的出处，以及从数据库资源中提取源数据时所用的标准或方法。

比如，如果建立一个蛋白质的同源性模型，那么就需要解释怎样选择模板的结构作为基础来建立模型。用未知序列去查找PDB会找到它们吗？那些方法提供充足的模板结构了吗？用以简图为基础的方法有助于鉴定与未知序列进化上关系较远的其他结构吗？进行的每个步骤都需要证明。

在计算生物学中，选择源数据系列的标准非常复杂。比如在序列信息的统计学研究或蛋白质结构数据的统计学研究中，经常选择可获得的蛋白质数据的非冗余(nonredundant)的子集。这意味着如果蛋白质序列与包括的其他蛋白质的系列太相似，这种个体的蛋白质数据将被排除在外。包括在微小差异的条件下，同样蛋白质结晶的两种结构数据文件对计算研究的结果可能会出现偏差。在自己的记录中或在已发表的源文献中，这样的选择过程每一步都需要证明。

记住所有的数字序列及结构数据都是派生资料。当你得到它时，起码经过了一至两个步骤，每一步骤都可能导致错误。DNA序列已经经过basecalling软件并汇编成图谱，错误分析及根据结构和功能进行注释等过程，所有的过程都是使用和你一样易犯错的科学家开发的工具进行的。蛋白质结构坐标只是一个很好的猜测，原子和观察到的根据X线反射的模式推算的电子密度数据相吻合。这并不是说你不应该使用或者相信生物学资料，但是应该记住在序列中与结构中原子坐标相关联的每一个看起来明确的性质都有一些不确定性。结晶学家提供参数，比如R-因子及b-值，在

某种程度上定性了大分子结构中坐标的不确定性。但是对于序列来说，在数据文件中不进行常规评估。

鉴定成功的标准

对结果进行严格的评估是建立生物学中有用的计算机模型的关键。在可以使用的现存的各种工具中，我们具有讨论评估结果的多种方法，如使用 BLAST 的 E值，从结果的长列表中挑出有意义的匹配以及评估蛋白质结构模型的几何学。在开始计算分子性质或开发计算模型前，列出你所知的目录，并进一步查找信息，然后尽力找到可以证明结果有效的信息的方式。这是将问题化整为零的一部分。

计算方法通常总会产生结果。它不像波谱，如果在样品中没有显著的物质，就得不到任何信号。如果进行一次 BLAST搜索，几乎总会得到一些结果，你需要知道怎样区分有意义的结果及垃圾结果，这种垃圾结果使你不停地将苹果与桔子进行比较（或比较过氧化物歧化酶和乙醇脱氢酶）。如果计算一个蛋白质分子的物理化学性质，或者模拟一条生物化学途径，你得到了一个充满数据的文件，最好的可能评价结果的办法是将它们与一些实验结果进行比较。在应用一个计算方法之前，先决定怎样评价结果，如果认为方法成功，它们需要符合什么样的标准。

进行计算实验并文档化

对计算课题的结果进行管理时，应该区分开原始结果以及后继分析的结果。在结果的记录中对资料进行的任何分析步骤都应该包括在一个独立的部分中（例如，统计检验和曲线拟合的结果）。这部分内容应包括你得到的有关数据及数据集合的任何观察值。将两部分结果分开记录，一部分是从实验过程中得到的数据，一部分是从收集到的数据进行深入研究得到的分析结果。

一般来说，对于计算生物学软件用户来说，一个趋势是保存阳性结果的数据与注释，而忽略对阴性结果的证明。即使已经使用基于 Web 的 BLAST 搜索了一个数据库，即使什么也没发现，这个结果也是信息。如果已经写出或下载了一个预期能做某事的程序，但是却不能完成，这个信息也是有价值的 —— 它对于继续进行你的课题的人或者想花费时间去推算出哪部分可以运行哪部分不能运行的人来说是有价值的。

计算生物学的文档问题

许多研究人员，包括那些在计算机上做所有工作的人，都保存着实验室记录本，它仍然是科学研究的标准记录装置。实验室记录本为实验室活动提供了翔实的记录，以这种形式保存实验室记录仍然是许多研究认可的条件。

然而，实验室记录对于你或其他人重复或继续研究来说，是一个信息不方便的来源。实验室记录是像流水帐一样编写的，只按日期分类。它们不能索引（如果你有许多空闲时间还可以，而大多数的研究人员不能做到），除了坐下来从头到尾进行查找这个不明智的方法以外，不能对一个主题进行信息查找。

计算机文件系统为清楚而详细地组织研究记录提供了直观的基础。课题每一部分的信息可以在同级中存在，而不是按顺序存放。除了在书架上有记录本之外（也可以不要），还可以在数据中存放电子记录。如果文件进行系统命名，并且在建立电子记录时应用简单的惯例，如 Unix 工具中的 *grep* 命令允许通过输入一个特定的单词或日期来查找文件，可以用比通读一个记录本快很多的速度找到所需要的信息。

电子笔记本

使用土方法建立工作的电子记录时，你可能会想用一个可获得的商业产品。或者，如果你正在寻找一个免费软件，来对电子笔记本概念进行补充，你可以得到由 DOE2000 电子笔记本计划开发的软件。 eNote软件包允许输入文本、上载图像和数据文件并创建图表和注释。这是一个 Perl CGI程序，在安装了 Web服务器及 Perl 编译器的任何平台上均可运行。安装时，可以在你的机器上从一个 Web URL进行访问,通过 Web形式上载笔记本。随着科学研究步入计算机时代， DOE计划的设计可满足实验室记录的需要。

eNote 软件包非常易于安装。它需要在机器上安装一个可以运行的 Web 服务器。如果你已经安装了的话，就可以下载 eNote 的压缩包，并在它的目录下打开，如 */usr/local/enote*。三个文件 *enote.pl*、*enotelib.pl* 和 *sketchpad.pl*都是 eNote 程序。需要将*enote.pl*移到 */home/httpd/cgi-bin* 目录下（或者在可执行 CGI 目录中,在 Red Hat Linux 系统中，这个目录是默认的），并重新命名为 *eNote.cgi*。如果想要限制访问记录本，那么针对 eNote 建立一个特定的目录并记住这个目录，可显示访问 CGI 脚本的 URL 途径。*sketchpad.pl*文件也应该移至这个目录，但不需要重新命名。将目录

gifs 和 *new-gifs* 转移至一个可通过 Web 访问的位置。你可以为此目的建立一个如 */home/httpd/enote* 之类的目录。解压缩时，不管文件 *enotelib.pl* 和目录 *sketchpad*。

最后，需要在 *enote.cgi* 和 *sketchpacl.pl* 中都编辑第一行，以找出机器中 Perl 可执行文件的位置。编辑 *enote.cgi* 脚本来映射安装 eNote 脚本及其支持文件的路径，也可以选择一个目录将 eNote 写进去。比如，你想创建一个 */home/enote/notebook* 目录并在那里保存 eNote 文件。如果是那样的话，可以肯定目录对其他用户可读可写（通常以用户 *nobody* 来鉴定），所以 Web 服务器可以在那里写。

eNote 脚本也包含特定的参数来确定笔记本的用户是否加、减及修改条目。如果你计划严格地使用 eNote，有许多重要的参数要考虑。你允许用户撕掉不想要的实验记录页码，或者是写得太多以至于原始记录不能阅读吗？ eNote 允许你在用户处理资料时保留控制权。

eNote 界面是一个简单的 Web 形式，也可以链接到 Java sketchpad applet。如果你想让在你的计算机上登录的特定用户访问 eNote CGI 脚本，可以在 CGI 目录的 eNote 子目录中建立一个 *.htaccess* 文件。*.htaccess* 文件是可由 Web 服务器读取的文件，包含限制访问某一特定的目录或可以访问的某位置的命令，关于建立 *.htaccess* 文件的更多信息，查询你使用的 Web 服务器的文档 —— 在大多数 Linux 系统中可能是 Apache。

如果开始使用电子笔记本来存储实验室笔记，记住要经常备份记录本以防系统出错。

第二部分

生物信息学工作站

- 第三章，建立工作站
- 第四章，Unix 中的文件和目录
- 第五章，在 Unix 系统下工作

第三章

建立工作站

在本章中，我们将讨论如何建立运行 Linux 操作系统的工作站。由于 Linux 是一种自由的、开放源代码的 操作系统，因此才有可能使普通的 机成为一个功能强大的工作站。使用 或者其他开源软件配置自己的机器之后，就可以用较低的费用运行大量功能强大的计算生物学和生物信息学工具。

在编写本章的过程中，我们遇到了一个两难的问题 为了能够自由漫游于 世界，首先需要在计算机上安装 系统；但是，在计算机上安装 系统之前，还要了解一些 知识。如果读者以前没有使用 的经验，我们强烈推荐，在建立自己的 Unix 工作站之前，首先浏览一下第四章和第五章。而如果你已经对 Unix 的输出和输入非常了解的话，可以跳到第六章。

在 Unix 操作系统下工作

大多数人可能已经习惯于在个人计算机上工作。他们可能对窗口界面、字处理程序甚至一些数据分析软件包非常熟悉。但是，如果你想把计算机当作自己科学研究中的一个重要成员，就需要工作在 Unix 操作系统下，或者工作在其他类似的多用户操作系统下。

操作系统能做什么？

没有安装操作系统的计算机就像没有生命的生物一样。安装操作系统之前，计算机不能做出任何反应，也不能起任何作用；它只能静静地“蹲”在那儿，瞪着一双无神的眼睛望着你，直到它走向腐朽（或者生锈）。安装操作系统之后，仿佛给计算机注入了生命，使之可以自由呼吸。操作系统进行一些低层操作，例如，它可以使硬件与之一起工作；提供一个舒适的环境，使你可以在其上运行和开发程序。由此可见，操作系统最重要的功能是使你能够舒适地生活和自由地编程。

为什么要使用 Unix？

既然操作系统是不需要给予太多关注的东西，那我们为什么又要操心选择使用何种操作系统，并且又要选择使用 Unix呢？

对于多用户计算机系统而言，Unix 是一个功能强大的操作系统。它已经存在了 25 年之久，主要应用于工厂和高等学校，因为在这些地方需要网络化的系统和多用户高性能的计算机系统。在最近几年，为了能成为用于个人计算机的操作系统，Unix 不断进行优化；即使那些没有应用到个人计算机上的Unix操作系统，也不断得到优化：增强了计算机之间的网络功能；初始化多个异步工作；保留着独一无二的支持多用户工作环境的信息；保护个人用户存储的信息不为系统中的其他用户接触。Unix是用于万维网的操作系统；支持网络的强大软件系统是在 Unix 环境下开发出来的，并且大多数服务器也是运行在 Unix服务器下。

因为 Unix 已经得到广泛的应用，并且在应用过程中开发出了许多运行在Unix系统下的科学数据分析软件，所以不用担心找不到大量应用在Unix系统下的高质量、有意义的科学软件。计算机生物学研究人员和生物信息学研究人员特别喜欢开发应用在 Unix 系统下的软件，这是因为在20 世纪中期之前，只有 SGI 和 Sun公司的 Unix 工作站能够实时地可视化蛋白质结构数据。

Unix系统有丰富的命令和应用程序。Unix的每一次发布都提供了功能强大的内置程序集。从网络软件到字处理软件，从电子邮件系统到新闻组阅读器，都已经成为Unix系统的一部分。同时，其他许多程序也可以免费下载并安装在Unix 系统上。

学习 Unix系统并在其上工作好像是一件遥不可及的事情。但是，事实上学习 Unix

命令的部分子集和成为一个有效率的Unix用户，并不需要了解或者使用 Unix 的每一个程序和特点。

不同风格的 Unix

Unix 并不是一个单调的实体。早就出现了许多不同的 Unix操作系统，它们有些是商业性的，有些是免费分发的，在本书中我们使用的 Unix 命令在你所能遇到的任何一种 Unix 系统下都以相同的方式运行。

Linux

Linux（读成 LIH-nucks）是一种开放源码的 Unix版本，它的名称取自其原开发者 Linus Torvalds，当时他是芬兰赫尔辛基大学的一名学生。 Linux 最初是 Linus Torvalds个人开发的用于个人计算机的免费 Unix 系统，后来在广大业余爱好者的共同开发下，逐渐发展成一个产品。 Linux 第一次使得广大个人计算机用户也有机会接触 Unix 系统。

在本书中，我们之所以重点介绍Unix 是因为以下三个原因：首先，由于 Linux的出现,使得 Unix 成为一个低廉的（或者免费的，如果读者有耐心自己去下载并安装的话）操作系统；其次，使用 Linux 之后，长期以来 Windows 用户的信息孤岛 ——廉价的个人计算机变得惊人的灵活，成为非常有用的工作站。配置好的 Linux操作系统消耗的系统资源比个人计算机操作系统要少得多，这意味着如果为计算机重新配置成运行Linux 的操作系统，它们可以更加轻松地运行，因为 PC应用程序和操作系统对系统资源的需求越来越高；第三， Linux系统是开发软件的杰出平台，因此，在 Linux 平台上有用于计算机生物学和一般性研究的丰富的工具库。

因此可以想像，如果你的计算机上安装了 Linux 系统，工作就会更加顺心。 Linux 是一个完全免费的操作系统。难道使用 Linux就必须完全理解如何在自己的系统中正确配置它吗？在几年前，这种想法也许是对的，但是现在情况不一样了。作为一种异于微软的操作系统的选择，生产计算机整机的公司已经开始在其产品上预装 Linux 了。同时，还有大量公司以合理的价格出售各种版本的 Linux ,其中最知名的就是 Rat Hat Linux（红帽子 Linux）。应该说明的是，笔者本人也运行的是 Rat Hat Linux 。本书中的大多数练习（也就是在本书中出现的例子）也是基于Rat Hat Linux 的。如果你已经从这些公司的其中一个购买了 Linux，你将获得几张光盘，它们不

仅包括 Linux，还提供了大量其他的免费软件工具。在安装过程中还可以寻求相应的技术支持。

Linux 可以在你的计算机上运行吗？

虽然 Linux 起因是 PC 的类 Unix 操作系统，但是，现在各种 Linux 开发项目几乎支持所有可能的系统结构，这包括所有类型的 PC 机、新型和老式的 Macintosh（苹果）机、SGI 工作站、Sun 工作站、HP 工作站和其他高端工作站或者高性能的多处理器机器。因此，即使混合使用新型机和老式机，你也可以使用 Linux 来创建一个由兼容计算机组成的多工作站的网络。关于安装和 Linux 的详细介绍，请参考“建立一个 Linux 工作站”小节。

其他常见的 Unix 版本

另外还有许多 Unix 的版本。在其他版本中，最常见的免费版本是 BSD（伯克利软件发布版），最初是由加州大学伯克利分校开发出来的。对于个人计算机而言，还有各种商业版的 Unix 系统，比如，SCO Unix。另外，还有一些工作站的生产商在他们生产的机器上销售其自己的 Unix 品牌，通常，这些 Unix 品牌有其特性和小技巧。在这些 Unix 操作系统中，最常用的是 Solaris（Sun），IRIX（SGI），Digital Unix（Compaq），HP-UX（HP）和 AIX（IBM）。上面的这些操作系统并不全面，但是，它们基本上代表了在实验室和计算机中心所能见到的操作系统。

Unix 系统的图形界面

虽然 Unix 是一种基于文本的操作系统，但是现在它并不是你所想像的那样，只是在黑色屏幕上充满了闪烁的绿色或者琥珀色的字母。大多数 Unix 系统使用各式各样的 X 窗口系统。X 窗口系统对屏幕环境进行格式化处理，使用户可以同时打开多个窗口和应用程序。由于 X 窗口是可定制化的，所以用户可以使用许多类似 PC 操作系统的菜单条和其他小工具。在主机和网络计算机上的单个 Unix shell 都以窗口的形式打开运行，这样可以使用户充分利用 Unix 的多任务特性，同时也可以一次运行多个 shell 程序。除了 Unix shell 程序和工具外，还有许多应用程序也利用 X 系统和 X 窗口作为图形用户界面的一部分，使其在应用 Unix 命令行方式运行的同时，也可以使用图形用户界面的方式运行。

在大多数主要的Linux系统下含有的GNOME和 KDE桌面环境,使用户的装有Linux系统的计算机更像一台个人计算机。除非你还习惯于打开 shell窗口或者运行 Unix程序，否则工具条、可视化文件管理器和可配置的桌面看起来和 Windows 或者Mac的工作环境没有什么两样。

建立一个 Linux 工作站

如果你已经是一个Unix/Linux系统的用户，可以跳过这节，直接阅读后面的部分。

如果你曾经使用过 Macintosh或者 PC操作系统，建立一个 Linux工作站或者服务的最简单方法是到市场中购买一台已经预装 Linux系统的 PC。例如，VA Linux 提供了各种预先配置好的基于Intel Pentium 的工作站和服务器，并且可以根据你的需要，在 n种最流行的 Linux 版本中进行选择。

如果你正在寻找一个完善的、相对独立的生物信息学系统，可以了解一下 *Iobion*，它正由 Iobion 系统公司(*http://www.iobion.com* ）开发。它是使用开放源代码技术开发出来的、一种新兴的生物信息学网络服务器设备。 Iobion 是一种基于 Intel 的硬件系统，计算机需要预装 Linux、Apache Web 服务器、一个 PostgreSQL关系数据库、R统计语言和一个综合的生物信息学工具和数据库套件。该系统用作局域内联网或者整个因特网上的Web客户端的科学应用。这些应用包括下面一些工具：和微数组数据库结合应用的微数组数据分析工具、序列分析和注释工具、公共序列数据库的本地副本、共享生物学数据的对等联网工具和先进的生物学实验室工具。Iobion 改进并且遵守生物信息学中开放的标准。

如果你已经拥有一台计算机，下一个目标应该是购买一套打包的Linux系统，Red Hat、Debian或者SuSE都提供这些软件包。这些预先打包的版本有以下优点：它们有方便用户安装 Linux的图形界面，所有这些软件都被压缩进打包管理器（对Red Hat而言是 Red Hat打包管理器或简称为RPM）压缩格式或类似的易于解压缩的格式；并且它们通常包括大量的“解压向导”，因此，使用打包管理器从各安装盘中安装比手工安装更加容易。

也就是说，我们假定读者已经购买了类似于当前 Red Hat的版本。那么读者可能会问：如何根据自己的需要选择工作站安装、服务器安装还是用户安装呢？这些选项的意思是什么呢？

可以轻松地在机器上安装 Linux 系统，以使其完成一些在 PC 机或者 Macintosh上无法做到的事情。读者可以在自己的机器上安装一个 Web 服务器，如果仔细阅读一下手册的话，可以发现如何给自己机器的每一个用户都分配一块空间，允许他们创建自己的 Web 页面。你可以安装一台匿名的 FTP 服务器，这样具有 guest 账户的人可以 FTP 下来一些文件的副本，这些文件已经被你设置成公开的文件。你还可以安装一个 NFS服务器来允许其他计算机上的一些目录可被自己登录。和工作站安装不同的是，只有某些选项是用于服务器安装的。

如果你没有用过 Unix 服务器的经验，可能首先想到的是把自己的Linux 机器安装成一台工作站。在将机器安装成Linux 工作站以后，用户可用这台机器浏览因特网，但是却无法提供任何对外界用户的服务（因为用户也没有责任维护这些服务）。如果想冒一次险，可以选择用户安装。这一选项允许用户自行选择自己所需要的系统组件，而不是一般地把所有东西都安装到机器上。

安装 Linux

在本书中，我们不可能详细地介绍安装和运行 Linux的所有细节问题。因为这些远远超出了本书讨论的范围。在市场中有许多非常好的介绍安装和运行 Linux 的书籍，读者可以根据本书的参考文献有选择地阅读一些好书。在本节中，我们只是针对在安装 Linux 中出现的重要问题提出一些建议。

系统需求

虽然Linux 可以在大多数 PC硬件配置情况下安装，但是有些硬件配置也可能不支持 Linux。因此安装和运行 Linux 有一些最小需求。为了优化 Linux 的性能，用户机器的 CPU 应该是 80486 或者更高的处理器。大多数 Linux用户都有使用 Intel 芯片组的系统。如果没有这种系统，就应该注意 Linux 支持一些非 Intel 处理器的情况。即使出现问题，也没有多少文档能够帮助你解决出现在这些系统上的潜在问题。

为了优化性能，系统应该至小配置 16MB 的 RAM（内存)。如果你打算运算 X的话，应该好好考虑一下自己是否要添加更多的内存 ——也许 64MB 会更好一些。虽然 X 在 16MB 内存条件下运行得很好，但是内存添加到 64MB 会使机器运行得更快，并且如果添加的内存足够的话，还可以多打开一些窗口。

如果想把 Linux 系统安装成工作站的话，至少应该为其留出 600MB 的硬盘空间。如果想把 Linux 系统安装成服务器的话，至少要为其留出 1.6GB 的硬盘空间。你可能从来都没有用到过这么大的硬盘空间。因此如果你正在安装一个新系统，我们建议你购买尽可能大的硬盘。你不会因为做出这样的决定而后悔的。

在大多数情况下，你所使用的 Linux 版本带的安装工具可以自动地检测系统配置，但是，如果检测失败的话，就必须自己准备处理所需的信息。表 3-1 中列出了开始安装 Linux 时需要了解的配置信息。

表 3-1　安装 Linux 需要了解的配置信息

设备	需要了解的配置信息
硬盘	• 每一个硬盘的数量、大小和类型 • 哪一个硬盘是主硬盘、哪一个是辅硬盘，等等 • 每一个硬盘使用的是哪种适配器类型（IDE 或 SCSI） • 对每一个 IDE 驱动器而言，BIOS 是否被设置为 LBS 模式
RAM	• 已安装的 RAM 的大小
光驱	• 每一光驱使用的是哪种适配器类型（IDE、SCSI 或者其他） • 对使用非 IDE 和非 SCSI 适配器的光驱而言，光驱的配置和模型情况
SCSI 适配器（如果有的话）	• SCSI 卡的生产厂商和型号
网络适配器（如果有的话）	• 网卡的生产厂商和型号
鼠标	• 接口类型（串口、PS/2 或者总线接口） • 协议类型（Microsoft、Logitech、MouseMan 等） • 按键的数目 • 对于串口鼠标而言，应该连到哪一个串口
视频适配器	• 显卡的生产厂商和型号 • 显存（RAM）的大小

为了得到这些信息，你可能需要检查一下自己系统的 BIOS 设置，或者打开机箱看一下已经安装的硬件。可以查询一下相关文档或者咨询系统管理员来了解这些内容。

下面是三种最流行的 Linux 版本：

- Red Hat（*http://www.redhat.com/support/hardware/*）
- boot: Debian（*http://www.debian.org /doc/FAQ /ch-compat.html*）
- SuSE（*http://www.suse.com*）

在提供这三种版本Linux的公司的网站上，都有其对特定PC硬件配置支持情况的详细信息。从表 3-1 中了解到安装 Linux 需要注意的信息后，也可以查询合适的网站以了解自己的 PC 硬件配置是否支持 Linux。

对硬盘进行分区

Linux 在已分区的硬盘上才能最有效地运行。分区（partitioning）就是把计算机的硬盘划分为几个可独立工作的部分的过程。硬盘上的每一个分区都是一个相互分开的文件系统。分区后的硬盘提供了一些额外的保护措施，使得一个文件系统中的文件不受其他文件系统的干扰。如果你下载了一些很大的图像文件，才可能填满那些根目录所在的硬盘分区；除非把所有可以得到的用于基本的系统功能的硬盘空间都填满，才会使计算机不能继续应用。同时，如果其中一个硬盘分区崩溃，你不必重新格式化整个硬盘驱动器就可以修复这些问题，这样做也不会丢失保存在其他分区上的数据。

开始安装 Red Hat Linux 系统时，需要先拥有一个装有 Linux 根启动文件的软盘，并且在光驱中还要有 Linux 光盘。打开计算机时，马上就会看到一个安装界面，它提供了几种安装模式选项（译注 1）。在屏幕的底部，有一个“ boot: ”提示符。一般情况下，在分区的过程中你只需简单地按下 Enter键即可，然而，如果你正在使用的是一款新型的计算机，特别是如果使用的是笔记本计算机，就需要在“ boot: ”提示符后输入“ text”，然后再按 Enter 键进入文本模式的安装方式。除非当前的 Linux 版本不支持你的计算机的显卡，才不能使用文本模式的安装方式。

在下面的几个安装窗口中，分别选择合适的语言和键盘模式。终于，你要进入这样的一个窗口，用于选择安装 Linux 的不同方式：GNOME 工作站、KDE工作站、服务器或者用户自定义安装。在该窗口中，只能选择其中的一种用户工作站选项，并

译注 1：　当然必须把软盘和光盘分别放在软驱和光驱中。

且需要慎重地选择其中的一种。然而，我们建议使用自定义方式安装，这样能够更加清楚地了解自己安装了什么，以及这些部件都安装到了什么地方。

如果在某个单机上安装 Linux 的这台计算机和网络上的其他计算机没有连接，就可以避免把整个 Linux 安装进一个大文件系统的麻烦，当然，这需要自己决定是否这么做。但是，如果安装 Linux 的机器将要和自己实验室中的其他所有机器共享它的“*/user/local*”目录，就需要进行一些特别的分区。

在任何给定的硬盘上，我们可以有四个分区。分区可以采用下面的两种类型：主（primary）分区和扩展（extended）分区。在一个扩展分区中，可以拥有多个子分区。Red Hat 和其他商用 Linux 版本都有简单的图形界面，用于帮助格式化硬盘。更高级的用户可以使用 *fdisk* 程序实现精确的分区。可以参考本书参考文献中推荐的有关 Linux 学习的系列图书，以深入了解分区的过程和如何使用 *fdisk* 程序。

选择主要的打包的组件

在完成硬盘分区，选择分区的开始点并且完成其他一些配置步骤之后，需要选择一些打包后的组件进行安装。

首先，进入打包组件选择列表。我们将明确选择一些组件：打印机支持；X Window 系统；GNOME 方式的桌面或者 KDE 方式的桌面（我们喜欢 KDE）；邮件、Web 和新闻组工具；图形操作工具；多媒体支持；工具软件和网络化的工作站支持。如果正在安装软件（并且需要如此），则需要一些存在于开发包组件中的项（C、FORTRAN和其他一些方便使用的编译工具，可能还需要一些开发库）。也许还想安装 *Emacs* 文本编辑器和授权发布工具。根据使用的系统的不同，可以给相应的工作站支持商打电话寻求支持。

打包组件的其余部分为计算机添加了服务器功能。如果想把自己的计算机用作 Web 服务器，则需要添加 Web 服务器组件。如果想把自己计算机上的某些目录用作NFS 登录服务，则需要选择 NFS 服务器组件。如果计划创建自己的数据库，则需要把机器安装为 PostgreSQL 服务器。一般情况下，如果不了解这些组件包或者不了解如何使用这些组件，可能不必在这一步中安装这些组件。

如果考虑自己计算机的硬盘空间，则需要筛选每个组件包中的内容，并且删除不会用到的软件。例如，“Mail，Web 和 News”组件包中包含许多不同类型的软件，用

于阅读电子邮件和新闻组。记住，不要不加选择地把该组件包中的所有软件都安装上，而是合理地选择自己所需要的组件，而不安装那些自己不需要的组件。（除非想知道如何选择安装这些组件，否则我们在这里给出一些提示以简化安装步骤：可以很容易地配置 Netscape 浏览器来完成所有需要的邮件和新闻阅读功能。）如果安装的是 Red Hat 系统，需要检查在："Applications/Editor" 组件下是否选择了 *vim* 编辑器；在"Applications/Engineering" 组件下是否选择了 *gnuplot*；以及在"Applications/Publishing" 组件下是否选择了 *enscript*。不要担心在最初安装时没有安装什么或以后又要安装什么，这些都很容易实现。

添加其他有用的组件包

一旦在计算机上完成了一个基本的 Linux 系统的安装，可以使用 *kpackage* 命令（如果使用的是 KDE 桌面环境的话）或者 *gnorpm* 命令（如果使用的是 GNOME 桌面环境的话）方便地添加新的组件包。

为了编译一些我们将在后面几章中讨论的软件，并且为了扩展 Linux 工作站的功能，需要安装下面的一些工具。第一组是来自 Red Hat Linux Power Tools 这张光盘的工具：

R
: 用于统计学计算和图形的强大工具。它基于 S，并且包括一个高级编程语言和一个运行时环境。

OpenGL/Mesa
: 一个用于创建图形用户界面的开发工具包，它大大地提高了一些分子可视化软件的性能。

LessTif
: 一个用于应用程序开发的工具集。我们可能不会直接使用这个工具集，但是当编译本书后面章节中讨论的一些软件时，可能会用到该工具集。至少要安装主组件包和客户端组件包。

Xbase
: 另外一个工具集。

MySQL
: 一个用于较小数据集的数据库服务器。如果想从头建立自己的数据库，MySQL 是很有用的。

octave

一个类似于 MatLab 的用于数值计算的高级语言。

xv 一种多用途的图像编辑和转换工具。

xemacs

一个功能强大的基于 X Window的编辑器，它具有特殊的用于编辑源代码的扩展功能。

plugger

一个通用的支持很多格式的 Netscape 插件。

可以从 Web 上下载并安装下面的这些工具：

JDK/JRE（*http://java.sun.com*）

如果想使用在后面的第四章中讨论的 Jalview序列编辑器这样的基于 Java的工具，我们需要一个 Java开发工具箱和 Java运行时环境。该工具对于来自 IBM、Sun和 Blackdown (*http://www.blackdown.org*) 的 Linux 是免费的。Blackdown 还提供了一个用于 Netscape 的 java插件，该插件对于运行某些我们讨论的应用程序是必要的。

NCBI Toolkit（*ftp://ncbi.nlm.nib.gov/toolbox/ncbi_tools/README.htm*）

这是一个用于开发生物应用程序的软件库。在编译某些来自 NCBI 的软件时，需要用到该工具。

StarOffice（*http://www.staroffice.com*）

这是一个来自 Sun 公司的免费的综合办公产品软件包。它实现了 Microsoft Office 和其他我们所熟悉的办公产品软件包的大多数或者全部功能。

如何使软件运行起来

在前面的几节中，我们已经开始进行了一些研究工作，并且找到了一些自己所需要的生物信息学软件包。现在我们要做什么呢？

在 Web 上寻找 Unix 的应用软件时，我们会发现它是以很多不同的格式发布的。每种类型的软件版本需要不同类型的处理。它们中的有些很容易安装，就像在 Mac或

者 PC 上安装软件一样。另一方面，某些软件以比较基本的形式发布，因此需要我们的积极干预，以保证该程序的正常运行。为了使这些软件能够正常运行，我们需要手工编译它们，甚至需要修改这些软件发送到编译器的目录，这样程序才能在自己的计算机上得以运行。编译（ compile ）是把软件从可读的源代码形式转变为机器可读的可执行（ executable）形式的过程。编译器是实现上述转换的程序。

那些难于安装的软件不一定是不好的软件。它可能是一个来自某个研究小组的高性能软件，没有提供一个便于安装的安装工具。当这些软件变得越来越不常用时，我们仍然有必要了解编译软件的知识。

Unix 的 tar档案

有时，软件通常是用 *tar* 档案格式发布的，*tar*是“磁带档案（ tape archive）”的缩写形式。我们将在第五章中详细讨论 *tar*和其他的文件压缩选项。但不巧的是，这些档案格式是在因特网上发布 Unix 软件的最常用的方法之一。 *tar*允许你下载一个文件，其中包括开发者进行软件安装和如何将其解压缩到正确的子目录中的完整画面。如果在使用 *tar* 时应用了 *p*选项，甚至会保留文件的特权。这一点保证了，如果开发者已经把所有需要的文件都打包进了 *tar* 档案文件中，你就可以相对容易地编译该软件了。

使用 Unix 的*compress* 命令（用扩展名 *.tar.Z*表示）,或者使用 *gzip*（用扩展名 *.tar.gz* 或者 *.tgz* 表示）命令，*tar* 档案可以进一步地压缩。

二进制版本

软件可以以未编译的源代码形式发布，也可以以二进制的形式发布。如果让你可以进行选择，并且你不知道怎么选择这两种发布形式，就用二进制形式吧。这将会解决很多令人头痛的问题。

二进制的软件发布版本是经过预编译的，并且已经准备在你的机器上运行了（至少在理论上是这样的）。当你下载了用二进制形式发布的软件之后，你将会面临一大堆选项去选择。例如，下面的列表是在公共 FTP站点上提供的用于 BLAST 序列比对软件的内容。这里有几种形式的文件，每种用于不同的操作系统；如果想在 Linux 工作站上运行软件，就应下载 *blast.linux.tar.Z* 文件。

```
README.bls                   52 Kb    Wed Jan 26 18:45:00 2000
blast.alphaOSF1.tar.Z        12756 Kb Wed Jan 26 18:40:00 2000   Unix Tape Archive
blast.hpux11.tar.Z           11964 Kb Wed Jan 26 18:43:00 2000   Unix Tape Archive
blast.linux.tar.Z            9334 Kb  Wed Jan 26 18:41:00 2000   Unix Tape Archive
blast.sgi.tar.Z              14746 Kb Wed Jan 26 18:44:00 2000   Unix Tape Archive
blast.solaris.tar.Z          12724 Kb Wed Jan 26 18:37:00 2000   Unix Tape Archive
blast.solarisintel.tar.Z 10679 Kb Wed Jan 26 18:43:00 2000   Unix Tape Archive
blastz.exe                   3399 Kb  Wed Jan 26 18:44:00 2000   Binary Executable
```

下面是二进制安装的基本步骤

1. 下载当前的二进制文件。确保下载的时候使用的是二进制模式。下载并阅读介绍文件（通常是 *README* 或 *INSTALL* 文件）。
2. 按照介绍文件中的步骤进行。
3. 如果需要的话，建立一个新目录，并把下载的文件移到该目录中。
4. 使用 *uncompress*（如果是 *.Z*文件）或者 *gunzip*（如果是 *.gz*文件）对这些文件进行解压缩。
5. 使用 *tar tf* 来检查下载文件的内容，并使用 *tar xvf* 解开它。
6. 如果存在配置文件和安装脚本的话，运行它。
7. 如果需要的话，使用 *ln -s* 链接二进制文件到你默认路径的一个目录中。

RPM 档案

RPM 档案是一种新的 Unix 软件发布形式，最近逐渐流行起来。这些档案可以使用 *rpm*命令进行解压缩。Red Hat 打包管理器（ Package Manager ）程序已经包含在Red Hat Linux 发布版本中，当你安装 Linux 时，它也会自动安装上。它也可以免费从 *http://www.rpm.org* 获得，并且可以使用在任何Linux或者其他的 Unix系统中。*rpm* 命令可在你的机器上创建一个软件数据库，简化安装和更新，甚至可以允许你创建 RPM 档案。RPM 档案可以是源文件形式，也可以是二进制形式，但是除了需要正确选择二进制的问题之外，使用这两种安装方式是同样有效的。

（当我们介绍命令的时候，我们将介绍每个命令的命令行的格式 ——例如，“用法：man name” —— 并且描述我们经常用到的一些命令选项。）

用法： rpm --[*options*] *.rpm

下面是一些重要的 *rpm* 选项：

rebuild

从源 RPM 中建立一个包

install

从二进制 RPM 中安装一个新包

upgrade

升级现有的软件

uninstall（或者 *erase*）

删除一个安装好的包

query

检查一个包是否已经安装好

verify

检查在包中已经安装文件的信息

GnoRPM

Linux的最近一个包含 GNOME用户界面的版本也包括一个名为 GnoRPM 的交互式安装工具。可以从主 GNOME 菜单的 System （系统）文件夹中得到该工具。如果使用 GnoRPM光盘安装，则只需要插入该光盘，找到光盘，点击 GnoRPM 中的安装按钮，这时 GnoRPM 就会提供一个你尚未安装的光盘上的所有包的可选择列表。也可以使用 GnoRPM 卸载和更新包，确保整个包已经从你的系统中干净地删除。如果在完成包的安装前，需要一些支持包，而它们又需要你下载代码库或者其他软件，GnoRPM 会通知你。

源代码形式的发布版本

有时，你的系统无法得到正确的二进制形式的发布版本，并且没有 RPM 档案，这时你就必须由源代码安装软件。

源代码可能容易安装，也可能难于安装。容易安装的一般包括一个配置脚本、一个安装脚本，以及一个用于操作系统保存编译器指令的 *Makefile* 文件。

一个便于安装包的例子是 LessTif源代码版本。LessTif 是 OSF/Motif窗口管理器软件的一个开放源代码版本。Motif 是开发用于高端工作站的，并且每年的注册费用是几千元；LessTif 免费地支持许多 Motif 应用程序（例如多序列比对包 ClustalX 和非常有用的 2D 绘图包 Grace)。LessTif 版本解包后是下面的形式：

```
AUTHORS         KNOWN_BUGS      acconfig.h      configure       ltmain.sh
BUG-REPORTING   Makefile        acinclude.m4    configure.in    make.out
COPYING         Makefile.am     aclocal.m4      doc             missing
COPYING.LIB     Makefile.in     clients         etc             mkinstalldirs
CREDITS         NEWS            config.cache    include         scripts
CURRENT_NOTES   NOTES           config.guess    install-sh      test
CVSMake         README          config.log      lib             test_build
ChangeLog       RELEASE-POLICY  config.status   libtool
INSTALL         TODO            config.sub      ltconfig
```

在 Linux工作站上配置和安装 LessTif实际上是一个非常简单的过程。以超级用户身份进入系统，把源 *tar* 档案文件移动到 */usr/local/src* 目录下。解压缩这些文件。在已经创建的目录（例如， *lesstif* 或者 *lesstif.0-89*）中，输入 *./configure*。运行配置脚本需要一段时间；等运行完成后，输入 *make* 命令进行编译。编译过程需要几分钟；最后，编辑 */etc/ld.so.conf* 文件。增加一行: */usr/lesstif/lib*，保存该文件，然后运行 *ldconfig -v*，使得你机器上的 LessTif 库可以共享。

像 LessTif这样的复杂软件是由很多不同源代码模块组合而成的。 *Makefile* 命令使编译器知道如何把它们放在一起，以组成一个大的可执行文件。其他程序十分简单：它们仅有一个源代码文件，而没有 *Makefile* 文件，在编译它们时，只需要向编译器发送一行指令即可。你应该通知编译器该编程文件名使用的是什么扩展名。 C 程序文件通常是以 **.c* 为扩展名，FORTRAN 程序文件则使用 *.f* 为扩展名，等等。为了编译 C 程序，需要输入 *gcc program.c -o program* ；而为了编译 FORTRAN程序，则需要使用 *g77 program.f -o program* 命令。参考编译器的手册页或者程序的文档（如果有的话），可以帮助你了解编译器命令的格式和可带有的参数。

编译器能够把人类可读的源代码转换成机器可读的二进制代码。每种编程语言都有它自己的编译器和编译指令。有些编译器是免费的，其他则是商用软件。在 Linux 系统上，你所遇到的编译器是 gcc（GNU 的 C 和 C++ 工程编译器）和 g77（GNU的 FORTRAN 工程编译器）（注 1)。在计算生物学和生物信息学中，你可能会遇到使

注 1: GNU是自由软件团体（ Free Software Foundation）为了开发一个完全开放的类 Unix 的操作系统而实施的一个合作项目。严格来说， Linux 系统应该是 GNU/Linux 系统，它是在 GNU公共许可（ GPL）规范下发布的，而 GPL 就是由 GNU 项目开发的。

用 C、C++、FORTAN、Perl和 Java 编写的程序。使用其他编程语言的情况则比较少。在开放源码发布版本中，你可以找到所有这些语言的编译器或解释器。

难于安装的程序可能会表现为几种形式。我们所遇到的主要问题之一就是源代码可能会依赖于某个代码库，而该代码库在我们的机器上又没有安装。注意检查该软件所附带的文档或*README*文件，以确定为了保证该软件能够正常运行，是否需要其他的代码或库文件。

一个非常有用的、但有些难于安装的软件例子是 ClustalX ，它是多序列比对程序 ClustalW的 X 窗口界面。为了保证在 Linux 系统上成功地安装 ClustalX,首先需要安装 NCBI工具箱和它所包含的 Vibrant 库。为了创建 Vibrant库，需要安装 LessTif库，并且需要在机器上安装 XFree86开发库。

下面是从源代码安装任意软件包的基本步骤：

1. 下载该软件的源代码发布版本。使用二进制模式；压缩后的文本文件被编码。
2. 下载并阅读介绍（通常是 *README*或*INSTALL* 文件；有时在解开文件后，必须找到该文件）。
3. 如果必要的话，建立一个新目录，并把档案文件移动到该目录中。
4. 使用 *uncompress* 命令（对于 *.*Z* 文件）或 *gunzip*命令（对于 *.*gz*文件）解开文件。
5. 使用 *tar xvf*或者根据帮助提取这些文件。
6. 按照提示信息进行（我们好像已经介绍了该过程）。
7. 如果有的话，运行配置脚本。
8. 如果有 *Makefile* 的话，运行*make* 命令。
9. 如果不存在*Makefile* ,并且只有 *.*f*或*.*c* 文件，则像前面所介绍的那样，使用 *gcc* 或 *g77* 来编译它们。
10. 如果有的话，运行安装脚本。
11. 使用 *ln -s*（这通常是 10步的一部分，但是如果没有安装脚本的话，需要手工创建该链接）命令，把最近创建的二进制可执行代码链接到在你的路径中包含二进制代码的目录中的一个。

Perl脚本

Perl语言通常用于进行 Web应用程序开发，并且经常为计算生物学家所用。 Perl 程序（被称为脚本）的扩展名为 **.pl*（如果是进行 Web 应用程序开发的话，则扩展名为 **.cgi*）。Perl 是一种解释型的语言；换句话说，在运行 Perl 程序时，不需要编译它。而实际情况是，在 Perl 脚本中的每个命令都被发送给一个称为 Perl解释器的程序，它执行解释命令（注 2）。

为了运行 Perl 程序，需要在计算机上安装 Perl 解释器。大多数 Linux 版本包含并自动安装 Perl。Perl 的最新版本通常可以从 *http://www.perl.com* 站点下载，该站点还提供了大量的帮助信息，介绍如何使用 Perl 。我们将在第十二章中讨论 Perl 语言的一些基本概念。

将其放到路径中

给出一条命令时，系统将在默认（ default）路径或在查找（ lookup）路径中查找其对应的程序（它也可以认为是可执行文件）。为了使自己更轻松些，在链接那些已经编译到 */usr/local/bin* 目录中的二进制可执行代码时，在运行它的时候，不必每次都键入该执行文件的整个路径名。如果跨文件系统链接，使用 *ln -s* 命令（将在第四章中介绍）就可以把命令链接到可执行文件的目录中。在运行程序时，链接的结果有时可能会造成“符号链接的层次太多（too many levels of symbolic links）”类的错误提示。在这种情况下，必须直接访问可执行文件，或者使用 *mv* 或 *cp* 命令把实际上可执行的文件移动到默认的路径下。如果这样做的话，应该确保也移动了这些执行程序所需要的支持文件，或者在程序运行的目录中创建一个对它们的链接。

一些软件的发布版本可自动在合适的位置安装它们的执行文件。通常使用的命令是 *make install* 。在编译程序后，确保运行了这个命令。如果想得到有关符号链接的更多信息，可以阅读参考文献中提到的某本 Unix 参考书，或者咨询你的系统管理员。

注 2：现在也出现了 Perl编译器，它可以用于对 Perl 脚本进行编译，以产生二进制的可执行代码。这样可以加速程序的执行。

在多用户中共享软件

在向Unix 系统中开始安装软件之前，首先要做的事情就是找到共享软件和数据保存在机器上的什么位置。一般情况下，本地应用程序安装在 */usr/local* 目录下,而可执行文件安装在 */usr/local/bin* 目录下。如果在你的机器上 */usr/local* 被设置为一个独立的分区，那么在升级操作系统时，就不会覆盖本地的软件安装。

为每个研究小组保留一套共享软件是个不错的主意。由系统管理员进行的对一个程序或软件包的单个标准版本的安装，在每个组成员工作方式完全相同的情况下，能够保证他们都可以使用该软件。这样可以使麻烦的问题更加简单，并且使结果保持一致。如果一个用户所面临的问题是运行组中其他人员均可使用的软件版本，则该问题的焦点在于用户的输入，而不必费尽心机地试图证明该软件的本地版本是否已经被正确地编译。

在大多数情况下，每个用户不必在他自己的目录中拥有某个程序的副本。其中主要的例外情况是，一个用户为了自己的需要，在实际上是否需要修改一个程序。在没有对其进行彻底的测试之前，这样的修改不应该影响公共的、标准的程序版本。因此，那些修改程序的用户需要拥有他自己的程序源代码和可执行文件版本。

什么软件是必需的？

新的计算生物学软件不断涌现出来，但是经过数十年经验的积累，出现了一套统一的工具和方法。许多科学家熟悉那些用于序列分析的标准商用程序包（例如GCG）和用于蛋白质结构分析的软件包（例如 Quanta或者 Insight）。对初学者来说，这些软件包为各种工具提供了一个集成的接口。

用于序列分析的商业软件包集成了大量功能，包括映射和碎片组装、数据库搜索、基因探索、比对和多序列分析、 motif 识别和进化分析。但是一个缺点是这些软件包都非常昂贵。在有限的预算下，特别是对教育工作者和研究小组成员来说，很难购买商业软件包，并为保留每年的许可证支付费用（这也需要几千美元）。

一个和费用相关的问题是，许多商业软件包，特别是那些用于大分子结构分析的软件包并不能在用户的 PC上运行。由于以前仅仅在高端工作站上才能进行图形操作

以显示蛋白质的结构，因此，这些软件包都是为高端工作站开发的。虽然现在大多数家用计算机都配置了高性能的图形卡，但是商业分子模型软件的开发者并没有急于跟上这个形势。

虽然商业计算生物学软件包可能非常出色，并便于使用，但是在尖端方法开发方面似乎落后了很多年。开发某个商业软件包的公司往往为每种类型的工具都仅仅提供一种方法，在开发周期的特定阶段购买了该方法，集中力量将其转化为可行的商业产品，但是可能并不适时地向其软件包中加入新的方法。

但是从另一方面讲，通常当学术软件处于边缘学科时，它并没有得到完善的开发，并且难于安装。文档（那些描述软件的公开文章）可能都不存在。学术软件包中的图形用户界面通常也是粗糙的，初学者很难学习。

在本书中，我们采用的是“小本经营的科学”的方法。在第六、七、九、十和十一章中，我们将介绍基本技术和经常用于分析数据的免费应用软件的快速参考表。我们希望这些列表能帮助你了解自己需要做的事情，如何找到做这些事情的工具，以及如何把这些工具以最适合需要的形式放在一起。这种方法可以使你摆脱软件企业巨头的限制，而接触最流行的方法。

第四章

Unix 中的文件和目录

在前面的章节中，我们已经介绍了安装 Unix 的步骤，下面，让我们花些时间介绍如何在 Unix 环境中漫游吧。在本章中，我们将介绍基本的 Unix 概念，包括文件系统的结构、文件的所有权和在文件系统中漫游的命令及创建文件（注 1）的命令。但是，本章的另外一个重要内容是，我们应该注意组织自己的研究数据，这样自己和别人才能够有效地访问这些数据。

文件系统基础

不管是对 Unix 系统，还是对桌面 PC 系统而言，计算机文件系统基本上都是相同的。文件（file）的命名都指向计算机存储设备上的某个存储单元。每个文件名都是一个有起点和终点的离散对象的指针，这个对象可以指一个可执行的程序，也可以指一些源代码。目录（directory）或者文件夹（folder）可以比喻成将文件分类存放的一个容器。计算机文件系统被组织为层次结构，它有一个根目录。在根目录下包括很多子目录的分支，而子目录下面可能又包含着其他的子目录。

注 1： 在本章和第五章中，我们将介绍许多 Unix 命令。我们对这些命令及其选项只是做一些简洁的、粗线条的介绍，应该能够帮助读者快速开始工作，但是这些绝不是对 Unix 命令的详尽介绍。在参考文献中列举了一些经典的 Unix 书籍，读者可以参考这些书籍了解 Unix 命令的细节。

如果适当地应用系统的这种层次结构的话，将有助于组织和共享信息。就像早期生物学家发展起来的对物种的分类法一样，文件层次将按照从一般层到特殊层的结构组织信息。每次都应该把文件系统分割进子目录，因为在较大类的文件中进行分割是非常有意义的。

为什么要把计算机中的文件以一种系统、有序的方式组织呢？看上去提出这样的问题很奇怪，因为其答案很明显。然而，研究者和研究小组面临的一个共同问题是不能有效地共享信息。当研究小组中有的成员离开，而其他人需要接手工作时，信息管理的问题就会变得突出起来。

想像一下，和你一起工作的同事把他的所有书籍和论文凌乱地堆放在他的办公室里。忽然有一天，你的这位同事找到了一份新工作，并且需要立马离开办公室，没有时间收拾办公室里的资料。你的老板告诉你不能把这位同事的论文一股脑都扔了而不检查一下，因为可能还有一些有用的东西。由于你的这位同事没有组织他的资料或者对它们进行分类，所以你就得检查每一项资料，确定这些资料是否有用，然后决定该把它们放在什么地方。如果不麻烦的话，也要浪费一周的时间，所以这肯定会是一个累人的工作。

如果涉及的是计算机文件的话，这类问题会更大。首先，由于许多非常有用的文件都是二进制的程序，而不是文本文件，因此用户不能读取它们。因此，如果不能读取其内容的话，很难确定这些文件是什么。而其他类型的一些文件，例如大量的数据文件，也许并不包含有用的重要信息，虽然它们可以作为文件读取出来，但是已经不可能说明其目的了。

第二，计算机所包含的有用空间要比一间办公室的墙壁所围绕的空间大得多。随着磁盘空间越来越廉价，用户更容易共享一个文件系统，而不必在用完后即删除自己的文件。许多程序产生了大量的输出文件，如果没有磁盘空间太小的限制，以至必须在运行程序后删除这些输出文件，这样就会在短时间内产生大量的垃圾文件。

如何才能避免这类问题的出现呢？第一步就是要注意引起这些潜在问题的根源。必须了解哪一类程序和文件应该和别人共享，而哪一类文件应该放到自己的目录中。应该建立方便的数据文件和程序命名，并且坚持这种习惯。应该使自己的文件系统以一种易分辨的层次构造。应该随时观察自己的计算机系统上的硬盘空间，如果不需要频繁地读取这些文件，可以把这些数据压缩存储。应该在文件系统、程序和数据文件中创建自己工作的信息文档。

文件系统层次的实质意味着你总是有一个强大的针对自己工作的索引系统。这可能是针对基于计算机的研究工作而言的，或者就像你的合作者把他的所有图书和论文凌乱地堆放在办公室里一样。但是为什么要有一个工作索引呢？因为不必费很大的工夫，你就可以使用计算机的文件系统使工作变得更有条理。

熟悉目录层次

与现代的操作系统一样，Unix 系统的层次也是树状结构。你也许已经在 PC 操作系统中习惯了树状结构。打开一个文件夹，里边可能有一些文件和文件夹，其层次可以随用户的要求不断加深。树状结构有一个根目录，用 / 表示。在一个组织良好的系统中，每一个子目录中都包含文件和其他子目录，用于表示特定的主题或系统功能。

当然，实际上计算机中没有任何东西像一棵树。文件存储在各种媒体中 —— 通常情况下是硬盘，它是计算机中的一种可记录的设备。就像其名字所指的那样，硬盘其实是一个磁盘。你在 Unix 系统中所观察到的树状结构是一种简单的索引方式，它存在于那些磁盘或其他设备中，例如光盘、软盘和 Zip盘，甚至存在于一组网络计算机的每个机器的磁盘中。 Unix系统具有强大的网络功能，它允许连接在网络上的计算机设备登录到网络中的其他计算机上。使用这些网络功能，几个网络化的计算机的文件系统都可以加上索引，仿佛它们是一个大型的、无缝的文件系统一样。

通向文件和目录的路径

文件系统中的每一个文件都可以惟一地用文件名和一个路径的组合标识出来。可以在文件系统中用给出其全名的方式引用任何文件，其全名以 /开头，表示根目录，接着是一系列子目录（路径的组成元素），以文件名结尾。在某个 home 目录下的文件的全名或绝对路径（ absolute path）可以写作下面的形式：

```
/home/jambeck/mustelidae/weasels.txt
```

绝对路径描述了文件和根目录／之间的关系。路径中的每一个名字都表示前一级目录的子目录，／符号将目录名分开。

文件系统中的所有文件或目录都可以用它们的绝对路径命名，但是也可以用相对路径（ relative path）命名，相对路径描述它们和当前工作目录的关系。你所在目录中

的文件可以用给出的当前工作目录中的文件名惟一地标识出来。你的当前目录的子目录中的文件可以根据它们所属的子目录命名。从 *jambeck* 的家目录中，可以使用 *mustelidae/weasels.txt*' 惟一地标识 *weasels.txt*文件。这里没有使用前置的 / 符号就说明，这里是根据与当前目录的关系来定义的路径，而不是相对于根目录定义的路径。

如果想命名一个与当前工作目录相同层次或高一层次的目录，可以使用一种简便的方法。文件系统中的每个目录都包含两个链接：*./* 和 *../*，它们分别表示当前目录及其父目录（即子目录的上一级目录）。如果用户 *jambeck* 目前在 */home/jambeck/mustelidae/weasels* 目录下工作，他可以把 */home/jambeck/mustelidae/otters* 表示为 *../otters*。和 */home/jambeck/mustelidae/* 目录位于同一层次的目录的子目录应该表示为 *../../didelphiidae/opossums*。

另一种缩写命名的习惯常应用于流行的 *csh* 和 *tcsh* shell 环境中，它将 home 目录的路径缩写为 ~。这样，*home/jambeck /mustelidae* 目录可以表示为 *~/mustelidae*。

使用基于进程的文件层次

文件系统可深可浅，或者说可长可短。最好先制定一个组织自己文件的指导性规划。文件系统的每一层都应该和你自己执行计划中的每一步相关联。如果把一个大型项目中的复杂处理步骤的输出都塞到一个目录中，那么这个文件系统的层次就会显得太浅。然而，仅从一个数据对象中得到的几个分析结果的项目目录没有必要写成很多子目录。如果一个进程的几种输出结果的目录相互嵌套，或者说同一层进程的分析结果都放在子目录中，那么这个文件系统就太深了。如果你能够用目录来清楚地表示出自己的工作过程，那么别人也就很容易理解你处理结果的数据，你自己也容易记住这些数据。

就像在前面看到的例子一样，home目录可能包含几个目录，每一个目录都包含特定项目的数据和文档。这些项目目录中的每一个都应该以一种可以反映项目规划的方式组织。每一个目录应该包含和其中数据相关的文档。

建立方便自己工作的文件命名习惯

Unix 对文件命名几乎没有限制。文件名称可以包含除／或空字符（空字符的二进制表示为所有位都为 0）外的所有字符。然而，需要特别指出的是，有些字符，例如

空格、反斜线或＆等在命令行中有特殊的含义，用于命名文件时可能会引起问题。在大多数系统上，文件名的长度可以达到 255 个字符。然而，在对文件命名时，采用统一的方式命名而不用独特的命名方式是明智的。毕竟，大多数人记得更牢的是经常使用的一些命名方式，而对一些从 255 个字符中组合出的特殊字符，则容易忘记。

在对文件进行命名时，一个通用的习惯是：使用一个特殊的文件名，然后跟一个点（.），接着是标识该文件的文件类型的扩展名。

使用计算机开始自己的研究工作，并对自己的数据环境进行结构化处理时，需要开发出一套适合自己的文件命名习惯，更可取的方法是，找到一套已经存在的命名习惯，并在自己的整个项目中自始至终地使用它。其实，最令人烦心的事就是在旧数据集中不断搜索，并力图找到以几种不同的方式命名的相同的文件类型。你找到过所有相关的数据或结果吗？你所寻找的文件能命名为一个完全不同的名称吗？如果缺乏一个习惯命名方式的话，那么在出现意外情况时，就必须打开每一个不可识别的文件，并一个一个去检查其格式。下面的小节将提供一个详尽的例子，来介绍如何建立一个文件系统，它可以使你不必十分费力地去寻找一个你已经知道具体位置的文件。

遵守下面的一些规则，有助于建立自己的文件命名习惯：

相同文件类型的文件应该有相同的扩展名。

从相同的源数据中派生的文件，在其惟一的文件名中应该有共同的元素（译注 1）。

惟一的文件名应该包含尽可能多的关于实验的信息。

在不影响其惟一性的情况下，文件名应尽可能地短小。

在你的工作中，可能会遇到已经建立好的文件命名习惯。例如，如果处理蛋白质序列和结构化的数据文件，你就会发现具有相同格式的类似文件具有共同的扩展名。另外，你也许会发现自己小组中的其他人也建立了他们的针对特定类型数据文件和结果的文件命名习惯。因此，你应该力图遵守所有的文件命名习惯。

译注 1： 即文件名有共同的字符。

结构化一个项目：一个例子

下面，让我们来看一个设置文件系统的例子。在这个例子中，列举了我们在工作中用到的实际目录结构；只是为了保护隐私，将目录名进行了改变。在本例中，我们将使用一个目录来保存整个项目。

把文件系统看做是一棵家族树有助于理解，在该树中，把项目中的相关方面内容放在一个分支中。项目目录的顶层应该包括两个文本文件，用来解释目录和子目录中的内容。第一个文件应该包括项目的纲要，包括日期、所涉及的人员的姓名、将要研究的问题和与该项目有关的出版物。根据惯例，这类信息文件应该命名为 *README* 或 *00README*，在 *shards* 项目中，一个最小的 *README* 文件可能包括下面的内容：

```
98-05-22
Project: Shards
Personnel: Per Jambeck, Cynthia Gibas
Question: Are there recurrent structural words in the three-dimensional structure
of proteins?
Outline: Automatic construction of a dictionary of elements of local structure in
proteins using entropy maximization-based learning.
```

第二个文件应该是一个索引文件（常被命名为 *INDEX*），用来解释子目录的整个布局。如果实际上你还没有搜集这些数据，则应该对目录的简单框架有一个解释。例如，下面的文件层次结构：

```
98-03-22 PJ
Layout of the Shards directory
(see README in subdirectories for further details)
/shards
/shards/data
/shards/data/sequences
/shards/data/structures
/shards/data/results
/shards/data/results/enolases
/shards/data/results/globins
/shards/data/test_cases
/shards/graphics
/shards/text
/shards/text/notebook
/shards/text/reports
/shards/programs
/shards/programs/source
/shards/programs/scripts
/shards/programs/bin
```

这也可表示为图表形式，如图 4-1 所示。

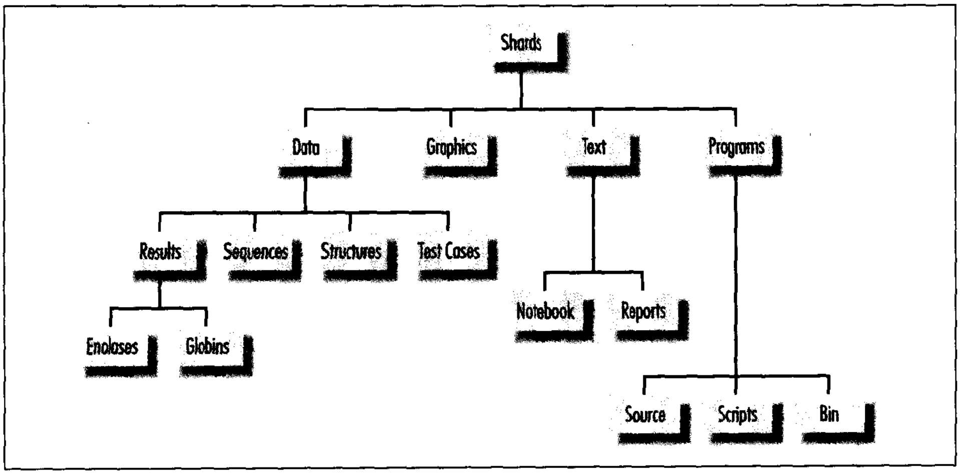


图 4-1　层次的树形框图

在这个目录中，我们第一次区分了程序（program）和数据(data)（程序包括我们写的软件，数据包括我们从数据库中得到的信息，或者程序产生的文件）。在每一个子目录中，我们进一步区分不同类型的数据［在本例中，分为蛋白质结构(structures）和蛋白质序列(sequences)]和运行程序作用于数据所产生的结果（运行在两组蛋白质中，即 *enolase* 族和 *globin* 父族），并对一些测试情况进行区分。程序又根据类型进行划分，即是人类可读的程序行（源代码），还是帮助程序运行的脚本，或者是二进制的程序文件。

就像我们在前面提到的那样，在文件中保存数据时，应该力图使用一个简短精炼和一致的系统命名文件。一旦需要使用 Unix 提供的工具进行自动搜索和更新文件时，如果使用精确地描述文件内容的非常长的文件名，而文件名又随文件类型的不同而不同时（例如，*all-GPCR-loops-in-SWISSPORT-on-99-7-14.text*）将会出现问题。在 *shards* 项目中，我们是从蛋白质数据银行（PDB）中得到的蛋白质结构开始的。然后我们使用了一个自己编制的 Perl程序 *unique.pl* 来产生一个非冗余的数据库，在其中任何蛋白质的序列和本组中其他蛋白质的相似性都低于 25%。这样，我们可以很简略地使用文件名 *PDB-unique-25* 来表示这个数据集中文件的信息。例如，在这个组中的蛋白质的名称列表和以 FASTA 格式（一种存储大分子序列数据的通用文本文件格式）包含蛋白质的序列的文件分别保存在下面的文件中：

```
PDB-unique-25.list
PDB-unique-25.fasta
```

包含派生数据的文件也可以统一命名。例如，包括来自非冗余集的所有 seven-residue 片蛋白质结构可以叫做 *PDB-unique-25-7.shard*。使用这种命名方式，如果想为这个非冗余集数据库相关的所有文件命名，可以使用通配符形式：*PDB-unique-25**，从而忽略由不同程序或在不同的类似阈值情况下使用 *unique.pl*产生的数据库。

在组织一个项目时，文件命名习惯只能起这些作用了；随着一个项目的增长，我们曾经所做的简单命名规划将会变得越来越复杂。对于较大的项目而言，应该考虑使用数据库管理系统（DBMS）来管理你自己的数据。在第十三章中，我们将介绍数据库的概念。

用于目录和文件的命令

到现在，大家已经具备了文件系统的基础知识，下面让我们继续深入学习 Unix中文件和目录的详细内容。在下面的几部分中，我们将介绍用于在文件系统中移动文件、搜索文件和目录，以及管理文件和目录的 Unix 命令。

在介绍这些命令时，我们将介绍每个命令（例如，“用法: `man` *name*”）所使用的命令行格式，并且详细介绍我们认为最有用的选项的一些运行结果。

在文件系统间移动文件

在 Unix 系统中打开窗口时，将会看到一个命令行提示符：

根据你的系统和 shell的配置不同，命令行提示符也是不同的。例如，下面是用户使用 *tcsh* shell 环境，并且已经配置了命令行提示符来显示用户名和当前工作目录的情况：

```
[cgibas@gibas ~]$
```

不管命令行提示符的风格如何，它的作用是计算机正等待你通知它做什么事。如果在提示符后键入一个指令，并按下 Enter键，就意味着你给计算机发出了一个指令。

Unix提供了一套简单的导航命令，以及用于搜索特定文件和程序的文件系统。我们将在第五章中更彻底地讨论命令的格式。在本章中，我们将介绍在Unix中游历的一些基本命令。

您目前所在位置：pwd

*pwd*表示“打印工作目录（print working directory）”，这就是它的实际任务。*pwd*为标准输出——将其打印在屏幕上发送出了你当前所在目录（即当前工作目录）的整个路径名。可以以下面的方式理解在某个目录中的含义：如果目录树是一个文件系统的地图，那么当前的工作目录就是“您目前所在位置”在地图上所指向的位置。

当你登录到系统中时，你的“您目前所在位置”指针会自动放置到home目录中。你的home目录是一个独一无二的位置。它既包括你每次登录到系统中时所使用的文件，还包括你所创建的用于存储其他文件的目录。如果你想了解自己的home目录在哪一方面和系统中的其他部分相关联，应该怎么办呢？在你的home目录的命令提示符下键入*pwd*命令，将得到类型下面的输出：

```
/home/jambeck
```

这表示*jambeck*的home目录是home目录的一个子目录，而反过来讲，home目录又是根（/）目录的一个子目录。

使用cd改变目录

用法：cd *pathname*（注2）

*cd*命令的作用是改变当前的工作目录。该命令仅有一个参数，通常在*cd*命令中表示一个目录的路径名。如果在不带参数的情况下使用*cd*命令，它将当前的工作目录修改为用户的home目录。

注2：当我们在第五章中介绍Unix shell和命令行时，你会发现，在命令行中Unix命令可以使用参数，也可以不使用参数。命令行中的第一个词通常是一个命令。接下来的词是参数，并且可能包括几个选项，它们修改命令的行为和操作对象，这些操作对象指出了路径名。命令行中的各词是由空白区域（空格或制表符）分开的各项。

为了使那些“您目前所在位置”工具更有用，首先需要以一种实用的方式来组织文件系统，这样，你所在目录的名称和位置就会给出相应的信息，指明哪种类型的材料可能在该目录中发现。安装 Linux 后，在默认情况下，机器中的文件系统的大部分已经安装好了，但是如何组织你自己的目录，即在哪里存储自己所使用的程序和数据，都是你自己的责任。

搜索文件和目录

Unix提供了很多搜索文件的方法，简单的方法是列出目录的内容，较复杂的方法是利用搜索程序来寻找特定的文件名和可执行程序的位置。

使用 ls 列举文件

用法：`ls [-options] pathname`

既然知道了自己所在的目录，那么如何才能发现周围有什么呢？只需在提示符下简单地输入一个 Unix列表命令 *ls*，就可以列举出在当前工作目录中的所有文件和子目录。也可以给 *ls* 命令一个目录名作为参数。那么，该参数会列举出在该目录下的所有文件名。

如果你有一个包括许多文件的目录，可以使 *ls* 加上一个通配符 *（星号）来生成一个特定部分的文件列表。使用 *的方法有几种。如果你的文件是一个系列（例如，*cb1* 到 *cb14*），或者文件具有共同的字符（例如，那些以 *.txt*结尾的文件），你可以使用 *立马把它们都列举出来。当 * 在命令行中作为参数出现时，可以表示文件名中的任意多个字符。例如，如果你想在一个有 400 个文件的目录中搜索名为 *seq11*、*seq25* 和 *seq34* 的文件，就不必用眼一行行地在文件列表中查找，可以使用下面的命令找到它们：

```
% ls seq*
```

如果想在同一个目录中搜索所有的文本文件，应该怎么办呢？我们知道文本文件通常以 *.txt* 结尾，因此可以使用下面的命令搜索文本文件：

```
% ls *.txt
```

使用 *ls* 时，也有各种命令行选项。最有用的一些选项如下所示：

-a 列出目录中的所有文件，即使这些文件前面有点号。默认情况下，*ls* 并不列举出以点号(.)开头的文件名，因此它们表示隐藏文件。隐藏文件通常包括用于程序的配置指令，并且，在有些情况下，必须检查或修改它们。

-R 一层层地列出子目录中的内容。列出当前目录的内容，并且无论到达哪一级子目录，其内容也会清楚地在目录列表中显示出来。该命令可以在文件系统中创建一个文件列表目录。

-l 一个很有用的选项，每行只列出一个文件名。只要使用正则表达式的技巧，源数据文件的单列列表就可以迅速地转变为一个 shell 脚本，在每一个文件上执行相同的操作。

-F 该参数包括一个指示文件类型的代码。如果文件名之后是 /，表示该文件是一个目录；* 表示该文件是可执行的；而文件名之后跟 @，则表示该文件是一个符号链接。

-s 以分块的方式和文件名一起列举出文件的大小。

-t 按文件最后一次修改的时间顺序列出文件。

-l 以长格式列出文件。

--color

使用颜色来区分不同的文件类型。

解释 ls 的输出

ls 以两种格式输出，即短格式和长格式。默认情况下是短格式。使用 *-F* 或 *-s* 选项时，仅包括每个文件的带有请求信息的名称，如下所示:

```
#corr.pl#          commands.txt        hi.c            psimg.c
#eva.pl#           corr.pl             nsmail          res.sty
#pitch.txt#        corr.pl~            paircount.pl    res.sty~
#wish-list.txt#    correlation.pl      paircount.pl~   resume.tex
Xrootenv.0         correlation.pl~     pj-resume.dvi   seq-scratch.txt
a.out              detailed-prac.txt   pj-resume.log   sources.txt
```

ls 命令的长格式输出包括各种有用的信息，包括文件的所有者属性和权限、文件大小以及文件最后一次修改的日期和时间，如下所示:

```
drwxrwxr-x    4    jambeck    weasel    2048    Mar5       18:23 ./
```

```
drwxr-xr-x    5    root      root     1024   Jan 20    12:13 ../
-rw-r--r--    1    jambeck   weasel   293    Jan 28    17:39 commands.txt
-rw-r--r--    1    jambeck   weasel   1749   Feb 21    12:43 corr.pl
-rw-r--r--    1    jambeck   weasel   559    Feb 23    14:52 correlation.pl
-rwxr-xr-x    1    jambeck   weasel   3042   Jan 21    17:05 eva.pl*
drwx------    2    jambeck   weasel   1024   Feb 16    14:44 nsmail/
```

上面的这个文件列表是使用 *ls -alF* 命令产生的结果。在显示的列表行中的前 10个字符给出了文件权限的信息。第 1 个字符描述了文件的类型。通常情况下，我们会遇到三种类型的文件：普通文件（用 - 表示）、目录（用 *d* 表示）和符号链接（用 *l* 表示)。

后面的 9 个字符实际上是由三组 3位字符组成的，它们包含文件的权限信息。文件类型后面的第 1 组 3 个字符表示用户的文件权限。第二组表示用户的组成员的权限，最后一组字符表示组成员外的用户的权限。字符串 *rwxrwxrwx* 的意思是，文件可以被任意用户读（*r*）、写(*w*）和执行(*x*)。我们将在介绍 chmod 命令时介绍如何改变文件的权限和文件所有者。

以长格式形式显示的文件列表的第 2列表示一个文件有多少个链接；也就是说，在文件系统中，这个文件存在于多少个目录列表中。这是因为，在多个目录中，可以命名相同的文件。在“管理文件和目录”一节中，我们将介绍如何为新文件和现有文件创建链接（目录列表）。

接下来的两列显示了文件的所有者。在上面的例子中，文件的所有者是 *jambeck*，它是 *weasel* 组的一个成员。

接下来的三列以字符形式显示文件的大小，并且显示了文件最后一次修改的日期和时间。最后一列显示了文件名。

使用 find命令查找文件

用法：find *pathname list* -[*test*] *criterion*

*find*命令是 Unix标准程序集中功能最强大、最灵活、最复杂的命令之一。*find* 命令基于各种参数来查找文件的一个路径或不同路径。*find*可以使用的参数有 20 多个，下面是一些最常用的参数：

–print

这个参数值总是 `true`，并且把当前文件的路径名发送给标准输出。 *–print* 应该在命令行的最后指定，因为它的值总是 `true` ,如果把它放到命令行序列中的其他参数前面的话，就会把被查找的路径名中的所有文件都发送到列表中。

–name

这是 *find* 命令中应用最多的参数，也是最有用的参数。 *find –name weasel.txt –print* 给出了在文件系统中名为 *weasel.txt* 的所有文件的完全路径名的标准输出。在基于文件名的情况下，可以使用通配符 *来匹配给定的字符串，以便查找文件。*find –name weas* –print* 命令不仅查找 *weasel.txt* 文件，而且还查找到了 *weasel.c* 和 *weasel* 文件。

–user uname

该参数查找由特定用户所拥有的所有文件。

–group gname

该参数查找由特定用户组所拥有的所有文件。

–ctime n

如果当前文件在 *n* 天前有改变，则该参数为 `true`。改变(change) 一个文件指的是对文件的任意改变，包括改变权限，而修改（modification） 文件则仅仅指对文件内部文本的修改。也可以使用 *–atime* 和 *–mtime* 参数检查文件的存取和修改次数。

在一个 *find* 参数的后面跟另一个参数相当于在这两个参数之间进行逻辑“与”操作。两个参数之间的 *–o* 表示逻辑“或”操作。反斜线（ /）是对一个命令求反，它的意思是只查找那些与参数不符的文件。

find 命令和其他命令结合起来应用，可以有选择地从一个文件系统中压缩或者删除特定的文件。如果想得到最近一周内在 home 目录和所有的子目录中修改的每个文件的列表，可以使用下式:

```
find ~ -type f -mtime -7 -print
```

如果把上式中的 type 修改为 *d* 的话，则仅显示新目录；如果把 –7 改为 +7 ,则显示在一周前修改的所有文件。现在，让我们回到最初的问题，并且查找可执行文件。使用 *find* 命令做上述工作的惟一途径是使用下面的命令：

```
% find / -name progname -type f -exec ls -alF
```

这个例子可查找出所有和 *progname* 匹配的文件，并且对每个匹配的文件执行 *ls –alF FullPathName*。任何 Unix命令都可以用做 *–exec* 的对象。清空 */tmp* 目录下的操作可以使用下面的命令来完成，通常情况下，该清空操作是由操作系统自动完成的：

```
find /tmp -type f -mtime +1 -exec rm -rf
```

上述命令用于删除在前一天没有被修改的所有文件。通常情况下，需要参考手册页来了解更多的细节（关于手册页的更多介绍，请参见第五章）。

使用 which 查找一个可执行文件

用法：which *progname*

*which*命令查找你的当前路径，并且如果在命令提示符下输入*progname*，就会报告该程序的整个路径。如果你想知道一个程序的位置，该命令很有用。例如，你想明确一下你正使用的程序的正确版本。如果程序不在路径所指向的目录中，则 *which* 命令不能查找它。

使用 whereis 命令查找可执行文件

用法：whereis -[*options*] *progname*

*whereis*命令查找一套用于可执行文件、手册页和源文件的标准目录。与 *which*命令不同，*whereis*命令不依赖于你的路径，但是它仅在有限的目录集中搜索程序，所以它不会给出程序位置的明确答案。

管理文件和目录

当然，就像整理桌子上的一堆资料那样，你需要周期性地整理你的文件和目录，以保持其清洁和整齐。Unix提供了用于移动、复制和删除文件的命令，同样也提供了用于创建和删除目录的命令。

使用 cp 命令拷贝文件和目录

用法：`cp -[options] source destination`

*cp*命令用于把源文件复制到某个目标位置。如果目标是一个目录，则源文件可以是多个文件，它们的副本将放到目标目录中。最常用的选项是 *-R*和 *-r*。它们都用于递归地复制文件，也就是说，它们把源目录及其子目录都复制到目标中。 *-R* 选项使 *cp*命令不能操作下面的符号链接；只有链接的本身可以复制。 *-r* 选项允许 *cp* 命令跟踪符号链接，并且复制它发现的所有文件。如果符号链接恰好在文件系统中形成一个循环路径，使用 *-r* 选项就会出现问题。

正常情况下，由 *cp* 命令创建的新文件可从 shell设置中得到它们的文件所有者和权限。然而，*cp* 命令的 POSIX 版本提供了一个 *-a*选项，它试图保持原始的文件属性。

使用 mv 命令移动和重命名文件及目录

用法：`mv source destination`

*mv*命令只是简单地把文件从源位置（source）移动或重命名到目标位置(destination)。文件和目录既可以是复制源，也可以是目标。如果复制源和目标都是文件，或者两者都是目录，*mv* 命令的结果实质上是对文件或目录的重命名。如果目标是一个目录，我们需要的是移动该层次的那个目录中的现有文件或目录，则在使用 *mv* 命令之前，该目录必须存在。否则，目标会创建为一个普通的文件，或者上述操作被当作是执行对一个目录的重命名。当复制源表示的是一个文件列表，而目标是一个已经存在的单个文件时，如果没有正确地使用 *mv*命令，就会出现问题。当问题出现后，复制源的每个成员都被重命名到目标中，然后迅速进行覆盖，只留下列表最后的文件不动。如果发生这种情况，只能找你的系统管理员，希望他能帮你找到最近的备份。

使用 ln 命令创建到文件和目录的新链接

用法：`ln -[options] source destination`

*ln*命令用于建立在目录树中不同位置的文件或目录间的链接。当创建链接操作在目标位置创建了一个新文件的快捷方式时，实际上并没有进行数据的复制，而是在文

件系统索引中创建了一个新指针，该指针允许源文件可以在“地图”上的多个位置被找到。

该命令最常用的是 *-s* 选项，用来创建一个对文件或目录的符号链接（或 *symlink*），正如下面的例子所示：

```
ln -s perl5.005_03 perl
```

该命令允许只键入单词 *perl* 代表其版本号，而不必记住 *perl* 当前版本的整个版本号。

ln 命令的另外一个常见用途是创建一个在系统路径（例如：*/usr/local/bin*）的一个目录下的最新编译的二进制可执行文件的链接。这样，如果想运行该程序，就不必指出其整个路径名。

使用 mkdir 和 rmdir 创建或删除目录

用法：`mkdir -[options] dirname`

用法：`rmdir -[options] dirname`

可以使用 *mkdir* 命令创建新目录，该命令只有两个命令行选项。

mkdir -p 创建一个目录和可能不存在路径的任何中间元素。例如，如果用户 *jambeck* 决定要在其 home 目录下创建一个目录 *mustelidae/weasels*，但是中间目录 *mustelidae* 不存在，*mkdir -p* 命令就会创建这个中间目录和其子目录 *weasels*。

mkdir -m mode 使用特定的文件权限模式创建一个目录。

如果目录是空的话，*rmdir* 命令可以删除它。使用 *-p* 选项时，*rmdir* 命令可以删除给定路径中的所有空目录。如果用户 *jambeck* 决定删除目录 *mustelidae/weasels*，并且除了 *weasels* 目录外，*mustelidae* 目录是空的，那么 *rmdir -p ~/mustelidae/weasels* 命令将删除 *weasels* 目录及其父目录 *mustelidae*。

使用 rm 命令删除文件

用法：`rm -[options] files`

rm 命令用于删除文件和目录。下面是该命令的常用选项：

- *-f* 不进行提示，强制删除文件。但是，你仍然不能删除你没有所有权的文件，你拥有的对文件的写权限会被忽略。例如， *rm -f a** 命令删除以字母 a 开头的所有文件，但是不会删除任何子目录。
- *-i* 在删除前提示 *rm: remove filename?* ,只有当你使用 *y* 或 *Y* 字符回答该问题后，才会删除文件。
- *-r* （recursive选项）删除文件列表中的所有目录和子目录。在这种情况下，并没有遍历符号链接，只有符号链接本身被删除。
- -v （verbose 选项）反馈被删除的所有文件／目录的名称。

虽然*rm*是一个非常简单的命令，但是有时对粗心的用户来讲也会造成严重的问题。

*rm** 命令删除目录中的所有文件。除非你的文件有些被设置成只读，或者设置了交互作用标志，你就会删除该目录中的所有文件。当然，这样使用的结果不会像使用 *rm -f** 命令或 *rm -rf** 命令那样糟糕， *rm -rf** 命令覆盖文件的任意只读模式，遍历你的目录，并且删除该目录及其子目录中的所有内容。

有些情况下，你将会发现在自己的目录下创建了一些奇怪的文件。例如，你可能有一个名为一*myfile*的文件，这里的 - 是文件名的一部分。如果试图删除该文件，会得到一个错误消息，提示 *rm* 命令并没有 *-m* 选项。shell 程序把 *-m* 解释为一个命令标志，而没把它当成文件名的一部分。对该问题的解决虽然有些琐碎，但并不总是会遇到这种问题，其方法是：只需要简单地提供一个对该文件的较完整的路径即可，例如，*rm ./-myfile*或*rm /home/jambeck/-myfile* 。如果在创建文件名时加入了一个空格，在删除该文件时也需要使用类似的解决方法。

在多用户环境中工作

Unix系统的设计目的是允许多个用户共享系统资源和软件，然而，同时也允许用户有选择性地保护自己的工作而不受别人干涉。为了在多用户环境下和他人一起工作，我们需要理解几个 Unix概念。

用户和组

如果使用 Unix 系统，你必须注册。以一个登录名标识自己，并且只有在输入了和登录名相对应的惟一的密码后，才能够登录系统。然后，你将控制文件系统的一片领地，在系统资源允许的情况下，该领地或大或小。你属于一个或多个组，并且可以和同一个组中的其他成员共享文件，而不需要使文件可让其他用户访问。在给定的时间内，你的组中只有一个是活动的，并且你创建的新文件会自动和活动组或原始组联系起来。如果你使用组权限和其他用户共享文件，并且你需要改变一个特定的组 ID，那么 *newgrp* 命令允许你改变你原始的组 ID。*id* 命令可以告之你的用户和原始组 ID 是什么。

账号信息保存在 */etc/passwd* 文件中，登录系统时，该文件可提供系统需要的信息。用户名和用户 ID 映射可以在这里找到，它还包括你的默认组、全名、home 目录和默认 shell 程序。在第五章中将介绍 shell 程序。你的密码的加密版本通常保存在这里，但是在大多数系统中，为了保密起见，实际的密码会从 *passwd* 文件中删除。其他的组信息可以在 */etc/group* 文件中找到。虽然在正常情况下它们是不能修改的系统文件，但是你仍然可以使用编辑器来查看这些文件的内容。

用户目录

当系统管理员创建一个新用户账号时，其过程包括：在 */etc/passwd* 文件中创建一个条目，在 */etc/group* 中也许会把你加入为一个组的成员，在系统的某个位置为你创建一个 home 目录，然后改变那个目录的所有权，这样你就可以拥有这个目录和在创建时放入的任何文件。在 */etc/passwd* 中的条目需要匹配你的 home 目录以及拥有你的 home 目录的用户和组的路径。当登录并应用在 *passwd* 条目中所列出的 Unix shell 时，你的 home 目录中也应该包括一组文件，这些文件设置了你的工作环境。在第五章中我们将详细讨论这些文件。

文件权限和状态

正像我们在介绍 *ls* 命令的那一部分讨论的那样，每一个文件和目录都有一个所有者，并且都有一个与其相关联的组。使用 *permissions* 创建的每个文件依赖于你的用户 ID 和组确定允许或拒绝你对文件的访问。在本节中，我们将讨论如何查看和改变文件的权限和所有权。

使用 stat 命令查看文件属性

用法： `stat -[optionns] filename`

stat 命令允许查看文件或目录的整套属性，包括权限、修改时间和所有权。这些信息可能比你需要的信息要多，但是如果你需要什么信息，都可以查到。例如， *stat image1.rgb* 命令返回下面的内容：

```
image1.rgb:
inode 11750927; dev 77; links 1; size 922112
regular; mode is rw-------; uid 12430 (jambeck); gid 280 (weasel)
projid 0  st_fstype: xfs
change time - Sun Mar 14 14:21:50 1999 <921442910>
access time - Sat Mar 13 18:11:21 1999 <921370281>
modify time - Sat Mar 13 10:28:39 1999 <921342519>
```

使用 chmod 改变文件的所有权和权限

在大多数Unix系统中,你并不想使所有文件都可以被所有用户读、写和执行。*chmod* 命令允许在一列文件和目录上设置文件权限或模式（mode）。递归选项 *-R* 会使 *chmod* 命令不断查找目录树中的下级目录，并且改变这些文件和目录的模式。

例如，对一个目录、一个符号链接和一个文件的长目录列表如下所示：

```
drwxr-xr-x  7  jambeck  weasel  2048    Feb 10 19:08  image/
lrwxr-xr-x  1  jambeck  weasel  10      Mar 14 13:12  image.rgb-> image1.rgb
-rw-r--r--  1  jambeck  weasel  922112  Mar 13 10:28  image1.rgb
```

在每一行的第1个字符都指出项目是一个文件、目录、符号链接，或者是Unix系统下的大量其他特定文件类型中的一个。在这里列出的三个是迄今最常用的。剩下的9个字符分别描述文件的模式。模式被分为三组，每组分别由三个字符组成。这三个字符以上面的顺序分别对应用户、组和其他。用户是拥有目录入口的账户，组可以是系统上的任意组，其他是指那些不属于包括用户和组的集合中的任意用户。在每个集合中，这些字符对应于那个人或组的读（*r*）、写（*w*）和执行(*x*）权限。

在前面的例子中，为了改变文件 *image1.rgb*的模式，以使得它仅可以被用户读取，并且不可以被任何人修改（写操作），可以使用以下命令中的一个：

```
chmod  u-w,g-r,o-r image1.rgb
chmod  u=r,g=-,o=- image1.rgb
```

```
chmod  u=r,go=-    image1.rgb
```

这些命令中的任何一个都会对 *image1.rgb* 的权限产生下面的结果：

```
-r-------- 1 jambeck weasel 922112 Mar 13 10:28 image1.rgb
```

前面的两个命令应该是很显然的。可以使用这种机制增加或减少用户的、组的和其他的读、写或执行权限。模式参数为：

[u,g,o]

用户，组，其他

[+, -,=]

增加，减少，等于

[r,w,x]

读，写，执行

u、*g*和 *o* 可以成组使用，也可以单独使用。 *r*、*w* 和 *x* 也是这样。+、- 和 =操作符描述将完成的行为。

使用 chown 和 chgrp 改变文件和目录的所有权

用法：chown -[*options*] *filenames item*

用法：chgrp -[*options*] *filenames*

*chown*命令允许改变文件或目录的所有者（或者，用文件权限的说法就是指用户）。*chown* 命令的操作依赖于你所运行的Unix版本。例如，IRIX 命令允许将所有权"给"其他入，但是在 Linux 系统下是不可能的。我们只能列举出 *chgrp* 命令的例子，因为在 Linux系统中，你可以是两个组中的一个成员，并使该命令可以为你所用。

chgrp 命令允许改变文件或目录的组。你必须是文件被改变到的组中的一个成员，因此你必须是几个组的成员，并且了解如何使用 *newgrp* 命令（这一点将在本章的后面谈到）。假定当你在默认组中时创建了 *image/* 目录，它是一个包含文件的目录。后来，你想和系统中的其他组的成员共享这些文件。因此，首先，其权限如下所示：

```
drwxr-xr-x 7 jambeck weasel 2048 Feb 10 19:08 image/
```

如果使用 *newgrp wombat* 命令改变到其他组中，则输入下面的命令：

```
chgrp -R wombat image
```

使得目录中的所有文件都可以被 *wombat* 组的成员访问。最后，你应该改变权限，使这些文件也可以由 *wombat* 组进行写操作。可以使用下面的命令：

```
chmod -R g+w image
```

这时，你的登记项应如下所示：

```
drwxrwxr-x 7 jambeck wombat 2048 Feb 10 19:08 image/
```

系统管理

在计算机上控制Unix系统配置的大多数文件只能被系统管理员进行写操作。增加和删除用户、备份和保存文件、在共享目录下安装新软件、配置 Unix 核心和控制对文件系统的不同部分的读取，通常是由一个特别指定的用户处理的任务，它的用户名是 *root*。在进行日常工作时，不应该以 root 用户登录系统，因为 *root*用户拥有普通用户所没有的权限，并且如果你拥有这些特权，你可能会把你的计算机系统搞得一团糟。从命令行中使用 *su* 命令临时性地应用系统管理特权，不仅可以完成那些必须由系统管理员所完成的任务，而且可以退回到你的正常用户状态。

如果你安装了一个 Unix 系统，你需要成为系统管理员或超级用户，并且需要学习各种必需的系统管理任务，来使自己的计算机保持在一个安全、有用的状态。幸运的是，已经有几本关于 Unix 系统管理员的参考书出版（有几本是由 O'Reilly 出版的），在每个 Linux 发布版本中，越来越多地包括更多的易于使用的图形系统管理工具。

用于组织文件的惯例

Unix 使用一套简单的名称用于描述在系统上发现的各种类型的文件。正常情况下，可以使用 *info*、*find* 或 *which*命令查找你所需要的文件，但是有时必须手工查找，并且你不愿意在 */bin* 目录中查找库。这些名称是在操作系统层使用的，但是它们也常用在子目录或软件发布版本中，用于把文件分开。

bin 可执行文件，或者二进制文件

lib 库，包括用于运行时或共享的库和那些在编译时需要的库。

spool

与外部设备和机器通信时，系统所使用的目录

tmp

临时存储

src 用于程序的源代码

etc 配置信息

man

手册页，文档

doc 文档

X X 或 X11R6 指 X 程序、库、*src* 等；通常目录具有一套非常完善的子目录集

一旦对如何组织和管理自己的文件和目录有了基本的理解，你就会很容易地理解如何在 Unix 环境下工作了。在第五章中，我们通过讨论最常用的 Unix 命令，来更清楚地介绍 Unix。为了能实际掌握 Unix 的技巧，我们强烈建议读者阅读参考文献中提到的一本或几本书。

在系统目录中定位文件

虽然你拥有的所有文件都应该在你的 home 目录中或其他特别指定用于用户共享的位置中创建，但是你仍然需要注意在系统的其他部分文件的位置。设计用于多用户的系统环境的一个优点就是，许多用户可以共享共同的资源，仿佛是控制自己文件的存取那样。

虽然我们说存在着一个标准的Unix文件系统有点不现实，但是像对待帕拉图的观点一样，我们将试图在某一个环境中把某一个系统想像为标准的。由于 Unix是由位于不同洲的几千个程序员开发的，它遵循 POSIX 标准，吸取了 Berkeley 和 AT&T 的 SysV Unix 系统的优点，因此，我们将使用 Linux 文件系统作为一个模板，并在必要的时候指出其主要的不同点。对 Linux 系统的当前标准的描述可在 *http://www.*

*pathname.com/fhs/*页面找到。在这里，我们给出了一个完整的文件系统的主要框架，并且指出了一些突出的特征。虽然在本节中描述的大多数目录仅可以由系统管理员配置，然而，作为一个用户而言，你有时可能需要知道在哪里可以找到系统文件和程序。图 4-2 列出了主要的子目录，在下面的列表中我们将对此进一步进行讨论。



图 4-2　Unix 子目录

/dev

包括那些将外围设备连接到系统上所需要的所有设备驱动程序。这些驱动程序用于 SCSI、音频、IDE 设备、PPP 和鼠标，其他大部分设备驱动程序也可以在这里找到。一般情况下，这里没有用户可配置的选项。

/etc

包括计算机中的所有配置文件。这些配置文件包括那些像系统名称、因特网地址、密码文件（除非你的计算机是某个大型机的一部分）、文件系统信息和 Unix 初始化信息。

/home

是 Unix 的一个常见的、但非标准的部分。 */home*通常是由一个非常大、被分割开的分区组成，它们包含所有用户的 home 目录。在 */home* 下有多个分割开的分区有利于使它在簇环境中共享，并且它也使用户很难完全填满一个重要的系统分区，以至于使其锁住。

/lost+found

系统目录是一个保存文件和目录的大仓库，但是有时会由于某种原因被系统放错地方。典型情况下，用户不能使用 *cd*命令进入这个目录。在系统崩溃或出现磁盘问题时，文件通常在 *lost+found* 目录中完成备份。有时，系统管理员通过简单地从*lost+found* 目录中移动一些文件并对其重新命名，就可以恢复那些看上去已经丢失的文件。在系统中的每个分区都有一个独立的 *lost+found* 目录。

/mnt

虽然在所有系统中都没有发现该目录，但是它是加载任意未被标准 Unix 文件系统描述的分区的典型位置。在 Linux 系统下，在该目录中可以找到一个加载的光盘或软盘驱动器。

/nfs

通常用作那些从远程机器加载的分区的任意加载点的顶级目录。

/opt

是对 Unix 文件系统的一个相对较新的补充。这是安装可选软件包的目录，这些软件包通常用于商业目的。在大多数系统上，你将会发现高端的、优化编译器安装在这里。

/root

root 的 home 目录，例如，对系统管理员而言是以 root 登录时的目录。

/sbin、*/bin* 和 */ub*

因为机器在没有 */usr* 分区存在之前，就需要开始启动进程了，所以任何优先加载到 */usr* 分区的正在使用该进程的程序必须驻留在主分区和 root 分区。例如，*/sbin* 目录的内容是 */usr/sbin* 目录的一个子集。标记目录 *sbin* 指出，只有系统级的命令才存在于此处，而那些正常用户可能不需要这些命令，因此也就不需要在他们的路径中包含这些目录。 */lib* 目录是系统库的一个小子集，只有在 */bin* 和 */sbin* 目录中的程序才需要它。当前的 Unix 程序使用 *shared* 库，这意味着许

多程序可以使用来自同一个库中的函数，因此该库仅需装载在内存中一次即可。这意味着在实际情况中，如果每个程序包括所有库例程的话，程序会占用尽可能少的内存；并且，如果正确的库已经被删除或者还没被加载，那么程序实际上也不能运行。

/tmp 和*/var/tmp*

一般情况下，这两个用户被配置为可被所有用户读、写和执行。像 vi等许多标准程序在运行的时候，都会向这些目录中的其中一个写入临时文件。正常情况下，系统有规律地自动清空这些目录，或者当重新启动机器时，也会清空这些目录。这里是写入临时文件的最佳位置，但是你不要指望系统将等待你来擦除它们。

/usr

该目录是存放用于 Unix文件系统的程序、编译器、库和文档的大仓库。当前为大多数 Unix 系统所建议的是，系统应该能够把*/usr*登录为一个相互分离的、只读的分区。在一个工作站簇机环境下，这意味着一个服务器可以输出一个*/usr*分区，并且在那个簇中，所有的工作站将共享程序。这将使系统管理员的工作较为轻松，并且可为用户提供一个有序的计算机环境。

/usr/local

这是安装程序和文档的典型目录，因此它们不会被操作系统重写。你在该目录中通常会发现安装了类似 Perl这样的程序，以及安装了其他一些从因特网上下载的软件。

/var

该目录为所有的系统程序所用，这些系统把其输出写到磁盘上。所有的系统日志、池和临时数据都被写到该目录下。包括登录信息，例如在启动过程中，由邮件程序、登录程序和其他所有系统进程所写下的信息。接收和发送的邮件保存在 */var/spool* 目录中，被当作文件发送给打印机。 *cron*、*batch* 和 *at* 命令所需要的信息也可以在这里找到。

第五章

在Unix 系统下工作

Unix 具有丰富的功能，在开始运行程序和搜集数据之前，需要注意这些功能中的一部分。在第四章中，我们讨论了在 Unix 中如何组织和管理你的文件，并且介绍了如何在文件系统中移动文件。在本章中，为了使大家能够更有效地工作，我们将带大家进入曲折的常用 Unix 命令旅程。我们将讨论 Unix shell 自身，在 Unix 中发布命令，如何从文件中浏览、编辑和提取信息， shell脚本，以及如何在多用户环境下工作。

一旦学会了这些 Unix 命令中的一部分，你将会发现它们的功能惊人地强大和灵活，仅仅使用通常的字处理程序只能以各种似乎不可能或者说不那么容易的方式修改文件，例如，使用一个命令，你就可以在 home 目录的所有文件中找到一种模式的所有实例。如果想使用一些简单技巧的话，你可以创建一个脚本，它可以等同地处理在你的源数据目录中的所有文件。使用另外一个简单的脚本，就可以在你每晚睡觉的时候，自动更新定制数据库的本地副本。

Unix Shell

当你登录进入一个Unix 系统或者在你的系统窗口管理器界面中打开一个新窗口时，系统会自动启动一个名为 shell 的程序。该 shell 程序会解释你输入的命令，为你提供在操作系统下工作的环境和界面。虽然当你在 Unix 系统下工作时，可能不使用 shell，而是使用图形化的文件管理器工具，但是你会发现，许多 shell命令对于数据

处理和分析还是很有用的。有些书通篇致力于介绍各种可用的shell，并且对一些常用 shell 而言，其手册页在打印出来后，会超过 100 页。我们在这里给你提供的仅仅是对一些通用的 shell的概括介绍，以帮助你学习。

这里用的是哪一种 shell

你所使用的 shell程序会影响命令行界面的使用。一些可以内置于 shell 程序的功能包括：一个简单的算术解释器，用于把命令行当做一个计算器；命令别名功能可以让你使用一个更方便的单词来代替那些标准的代表Unix命令的单词；文件名自动完成功能使你只输入几个必要的字符，就可以把目录中的一个文件和另一个文件区分开来，而不必输入整个文件名；命令编辑和命令历史功能可以使你回退到自己在前面发布的命令，并且在命令行中编辑它们；拼写检查功能；以及 shell程序的帮助功能。

在 Unix系统上有很多常用的 shell程序。当系统管理员为你设置账号时，会自动为你指定一个 shell。在 Linux系统上，默认的 shell程序是 *bash*（Bourne Again）shell。然而，你可能会更愿意使用其他的一种 shell来取代 *bash*。有两类主要的 shell程序：派生自 Bourne shell —— *sh* 的 shell，和派生自 C shell —— *csh* 的 shell。Bourne 类型的 shell 包括 *sh*、*bash*、*ksh*（Korn shell）和 *zsh*（Z shell）。C 类型的 shell 包括 *csh*和 *tcsh*。

由于历史原因，我们倾向于使用 C shell。当我们刚开始在 Unix系统下工作时，C shell是最好的 shell，而且*tcsh*程序已经把原来的 *csh*扩展为一个功能更强大的 shell。*tcsh* 实现了大多数我们想得到的 shell特性，包括命令历史记录、命令别名、文件名自动完成、命令行编辑、算术和函数、任务控制和拼写检查。*tcsh*也是一种最适合用户配置的 shell。因此，我们将从C-shell 的角度来讨论 Unix 命令的行为，假设你像我们一样也使用 *tcsh*程序。

你所使用的默认 shell 将会在你登录后的 */etc/passwd* 文件的最后一项列出。如果你不能确定当前使用的是哪一种 shell，可以输入下面的命令来得到它：

```
% finger your-user-name
```

对于用户 *jambeck* 而言，该命令会显示下面的信息：

```
Login name: jambeck            In real life: Per Jambeck
Directory: /home/jambeck       Shell: /bin/tcsh
```

这条信息告诉我们，该用户正在使用 *tcsh* 作为其默认的 shell。出于实际应用情况，我们将把讨论限制在 *csh* 和 *tcsh* 上，并且大多数情况下指的是 *csh* 和 *tcsh*。但是也会提到，默认情况下，许多系统进程（例如 *batch*、*at* 和 *cron*）使用的是 Bourne shell，因此我们也有必要至少学习一些该命令语言的基本知识。在大多数系统上，可以使用一些命令来改变在 *passwd* 文件中设置的默认 shell。如果使用的是 Linux 系统的话，使用 *chsh*（改变 shell）命令允许你改变自己的默认登录 shell。

在 Unix 系统上发布命令

发送一条指令到 Unix 上有其标准的格式。在本书中，我们将介绍命令和命令行。最初Unix的许多命令都有一个实实在在的可执行程序和它相对应，发布该命令就是通知Unix去执行这个程序。在本节和下面的小节中，我们将非常快地浏览一些命令及其概念。虽然我们可以针对那些最有用的 Unix 特性，给你一个简要概括，但是本书的写作目的并不是想取代一本复杂的 Unix 参考书。如果你是一个 Unix 的初学者，我们强烈建议你复习一下辅助书籍中的 Unix 基础，例如《Learning the Unix Operating System》、《Running Linux》（译注 1）或《Unix for the Impatient》。在参考文献中，我们也会提供一些推荐的读物。

命令行格式

命令行由命令本身、可选择的参数（修改命令如何工作）和操作数（例如命令在哪个文件上操作）组成。例如，我们刚刚简要讨论的 *chsh*（改变 shell）命令就有几种可能的选项。第一个是 *–s* 选项，它必须跟一个 shell 程序名作为其参数。第二个是 *–l* 选项，它不需要参数，该选项列出了在你的系统上可用的 shell。*chsh* 命令的操作数是那些 shell 被改变的用户的用户名。因此，为了改变默认 shell 程序，你可能会先输入：

```
% chsh -l
```

译注 1：中文版《Unix 操作系统》、《Linux权威指南》已由中国电力出版社引进出版。详情请访问：*http//www.infopower.com.cn*。

该命令给出了在系统上可用的 shell 程序列表：

```
/bin/bash
/bin/sh
/bin/ash
/bin/bsh
/bin/bash2
/bin/tcsh
/bin/csh
/bin/ksh
/bin/zsh
```

然后，为了实际把 shell 改变为 *tcsh*，应该输入：

```
% chsh -s /bin/tcsh yourusername
```

选项可以是一个简单的单字母代码，或者它也可能有自己的参数。没有参数的选项可以以组的形式给出，而带有参数的选项则必须分开指定。每个选项组或者分开的选项前面必须有一个连字符（ – ）。在一个选项组或分割选项中的最后一个选项可以跟选项参数。操作数紧跟在选项列表的最后一个选项后面。

坦白地说，许多 Unix 命令的选项你可能从来都没有用到过。因此我们不打算讨论这些选项。但是有下面的几种方法可得到 Unix 命令的选项信息。

Unix 信息命令

Unix 有内置的参考手册，它非常全面，并有指导性，它可提供关于命令的正确信息，并提供在你正使用的特定系统上，这些命令有哪些可用的选项。

man 命令是最有用的 Unix 命令之一；它允许你浏览 Unix 手册页。虽然一些 Unix 系统已经实现了一个 Web 浏览器形式的界面来显示 Unix 手册页，但是你也别指望总是能有这种浏览方式。而 *man* 命令则在所有类型的 Unix 系统下都是可用的。

用法：man *name*

这里 *name* 可以是一个 Unix 命令，例如 *grep*；或者是一个系统文件，例如密码文件 */etc/passwd*。

如果对自己正在寻找的命令没有把握，有时可以使用 *man* 命令的更短小、更智能的兄弟命令 *apropos* 来找到正确的信息。*apropos* 命令使用关键词查询来定位命令。

用法：apropos *name*

例如，如果你想了解自己系统上的磁盘使用情况，可以输入 *apropos usage*。如果你的 PC机上运行的是 Red Hat Linux的话，该命令的输出为：

```
du (1)  - summarize disk usage
getrlimit, getrusage, setrlimit (2)   - get/set resource limits and usage
quota (1)     - display disk usage and limits
quotacheck (8)      - scan a file system for disk usages
```

apropos 命令并不总是生成这么简要和介绍性的输出。获得你想到的输出的关键是，巧妙地输入各种关键词的组合（总是进行这样的搜索）。如果你想要一个可预知的 Unix 命令的列表，则最好翻阅一下某本全面的 Unix 图书。

如果在手册页中发现了下面的文字，你会怎么办呢？

```
This documentation is no longer being maintained and may be inaccurate or
incomplete. The Texinfo documentation is now the authoritative source.
```
(本文档不再进行维护，可能是不准确或不完整的。Texinfo 文档现在是权威的资源。)

Unix工具的 GNU 集（注 1）正在采用一个称为 *texinfo*的文档系统，它和传统的 *man* 系统不同。如果遇到这条信息，通过 *info progname* 命令，你应该能够阅读有关程序的最新文档。例如，*info info* 提供了一套关于 *info* 命令的使用的完整文档集，当你开始编写自己的程序时，甚至可以提供指令用于创建你自己的 *info* 文档。

标准的输入和输出

默认情况下，许多 Unix命令从标准输入中读取，并且把它们的输出发送到标准输出。标准输入和输出是与终端相关联的文件描述符。从标准输入读取的程序将会简单挂起，并等你在键盘上输入后，按下 Enter 键。负责把结果写到标准输出的程序把它的输出输出到终端，有时快得让你来不及阅读。

一些 Unix 命令把由空白区域围绕的连字符（ – ）读取为“来自标准输入的数据”。然后，这个概念就可以在命令行中用来替换一个文件名。如果没有输出文件名，程序就会将结果写到标准输出中。

注 1:　GNU 工具是由自由软件基金会的 GNU 工程发布和维护的。 GNU 代表“ GNU's Not Unix”，指的是由 GNU工程（*http://www.Gnu.org*）开发和维护的完整的类 Unix操作系统。

命令输入和输出的重定向

标准的输入和输出描述符是很有用的，这是因为，不用编写程序就可以把标准输入和输出与文件名相关联起来，从而重定向标准输入和输出。下面是最常用的 C Shell 使用的重定向语法：

- <　在文件名前面的重定向符把该文件名和标准输入关联起来，例如，文件的内容被当作是标准输入而输送给程序。
- >　这个重定向符把文件名和标准输出关联起来，这样在执行命令后，即创建了这个文件名；或者即使是已经存在这个文件名，使用输出的命令后，也会覆盖已经存在的这个文件。
- >>　这个重定向符把文件名和标准输出关联起来。它和＞的不同之处在于，该命令的输出是把结果添加到现有文件的结尾。

cat 命令读取一个文件的内容，并把它们写到标准输出中。如果想使用 *cat* 命令把三个文件的内容合并到一个新文件中，可以使用像下面这样的重定向命令：

```
% cat file1 file2 file3 > file4
```

例如，如果刚刚从 NCBI 网站上下载了一些独立的序列文件，并且想把它们集合到一个大文件中，使它可以被其他程序读取，那么可以使用 *cat* 的上面这种语法。（这是看起来比较简单的一个例子，实际上使用标准 PC 字处理程序很难处理，而且很费时间。使用 Unix 我们几乎不用打开任何文件就可以干净利落地解决此问题。）

也可以使用重定向符来把一个文件的内容作为标准输入导入一个运行着的程序（这种情况在运行程序 ,并提示你通过键盘输入时是有用的）；或者可以使用重定向符捕捉来自一个程序的输出 , 正常情况下该输出会被写入到标准输出中。这两种情况使用的命令如下：

```
program < inputfile
program > outputfile
```

例如，假定你刚刚结束一个扩展 BLAST 搜索，并且想把结果发送给你的同事。你可以使用重定向符 <（“ 小于 ”）把文件 *huge_blast_report* 从你的目录中快速获取，然后直接把它通过 mail 命令发送给你的同事，命令如下：

```
% mail colleague@university.edu < huge_blast_report
```

如果你想使你的同事能方便地了解该电子邮件的主题，可以使用 mail 命令的 *-s* 选项为该电子邮件添加一个主题信息，命令如下：

```
% mail -s "surprise!" colleague@university.edu < huge_blast_report
```

上述操作的反向操作，即把标准输出的结果（或者在屏幕上显示的文本）发送给一个文件，可以使用 > （“大于”）命令完成。也许你的同事想为自己做一个备忘，以提醒自己回复你的邮件。她可以使用 *cat* 命令来做该工作，以接收来自键盘的输入，并把它重定向到一个文件，命令如下：

```
% cat > reminder_to_self
Ha! Send fifteen BLAST reports to colleague on Monday
^D
%
```

Ctrl-D（^D）符号表示你已经结束了输入文本。在你同事的当前工作目录中，她拥有了一个被称为 *reminder_to_self* 的文件。

操作符

在对标准输入和输出的导向方面，操作符和重定位符的作用类似。然而，它们从命令的输出中得到另一个命令的输入，而不是像重定位符那样把输出和输出导到文件中。

最经常使用的操作符是管道（|)。管道把一个命令的标准输出导出，作为下一个命令的标准输入。这使得我们可以把几个不同的过滤命令或程序链接在一起，而不必每次创建输出或输入文件。

可以使用 *cat* 命令来引导一个文件的内容进入一个程序，该程序从标准输出读取信息：

```
% cat inputfile | program
```

该命令的语法可完成与前面我们所举例子（*program< inputfile*）相同的操作。这两种方法都把 *cat* 命令的输出作为程序的输入。如果想使用稍微不同的输入多次运行同一个程序，你可以创建多重输入文件，然后编写一个脚本，用来依次 *cat* 这些输入文件的每一个，并把它们的内容管道给程序。

不必写到磁盘上，管道就可以执行一套完整的文件处理选项。例如，想像你有一个数据文件，它由多个连接在一起的表组成。该文件中的第一个表占用了最初的167行，第二个表占用了下面的 100 行，文件的其他部分由第三个表占用（注 2）。你想得到中间表的第二栏所包含的信息，在该行中的字符为从 30到 –39。使用过滤器和管道，可以构造下面的命令来显示自己所要的数据：

```
% head -167 protein1.pka | tail -100 | cut -c30-39 > protein1.pka.data
```

在这个例子中，*head* 发送一个指定文件或几个文件（在此例中是 *protein1.pka* ）中的顶部的 167行到标准输出中；*tail*得到了 head的输出结果的最后 100 行；*cut* 命令从 *head* 和 *tail*命令的结果中得到了正确的字符栏，并把它存储在 *protein1.pka.data* 中。

通配符

一个有用的、Unix shell 认可的语法为在文件名中使用通配符。在把文件名传递给程序时，shell 为任意的通配符定位匹配对象。两个最常用的通配符是星号（ *）和问号（ ?）。* 表示“除了／字符之外的任意的 0 个或多个字符的序列”。?表示“任何单个字符”。这样，“本目录中的每个文件”均可以用一个 * 表示出来，非常方便。

shell 也认可其他的通配符。语法 [ ] 表示在一个特定集合中的任意字符。如果想把所有以字母 *m* 开头的文件都移动到一个新目录中，可以使用命令： *mv [a -m]* ../newdir*。如果想将以数字开头的所有文件都移动到一个新目录中，可以输入命令：*mv [0–9]* ../newdir*。

运行 X 命令

在运行 X 窗口系统的 Unix 系统上，有很多命令可启动带有非命令行功能的程序。一旦这些程序（从图形查看器到复杂的科学应用程序）可以从命令行调用，它们使用 X 窗口系统打开它们自己的窗口，其窗口一般包括一个完整的、独立的图形用户界面。

注 2：　这并不是完全虚拟的格式。它与我们经常进行的计算（蛋白质中某个氨基酸的 pKa值）的输出文件格式很接近。

查看和编辑文件

你可能已经熟悉使用程序来打开文件的方法。或者，如果在过去的 15年中,有一天你第一次使用计算机，你可能会习惯于简单地点击一个文件图标就打开一个文件，该图标通过打开文件的正确的程序段识别。

在 Unix中设计了一些命令，在需要的时候，可以用这些命令来打开像文本文件那样的可读取、可打印的文件。因此，文本文件可以由各种各样的命令打开，这些命令使得在操作文件时有很高的灵活性。文件读取和处理命令有下面一些功能：在文件的每一行，基于一个特定子字符串的值对数据排序；删除一个文件的特定部分；把数据栏并排放在一起；检查两个文件有何不同；在一个文件或文件组中搜索某些东西。通常，这些简单的命令都是你所需要的，用于在一个文件中提取你想得到的数据子集，并把它准备用做分析。

Unix有很多方式可以用来查看和编辑文件的内容。有些用于文本和程序的查看器允许检查二进制文件的内容，还有些具有全功能的编辑器用于修改无格式的文本文件。

使用 cat查看和合并文件

用法：`cat -[options] files`

*cat*命令可将文件的内容转储到屏幕上。如果文件比较短小，或者如果你已经成功地结束了一堂速读课，这个工具将工作得很好。然而，如果你需要查看文件的每一页是什么，那么 *cat* 命令的作用就不明显了，因为文件内容的滚动是不停顿的。

除了查看文本外，*cat*命令另一个最有用的作用是合并（或者说是连接）文件。例如，如果程序输出一系列文件，分别命名为 *meercat1.txt*、*meercat2.txt*和 *meercat3.txt*，如果你想把它们合并为一个文件的话，可以输入下面的命令：

```
% cat meercat1.txt meercat2.txt meercat3.txt > big-meercat.txt
```

这个命令将把 *meercat3.txt* 文件的内容添加到 *meercat2.txt* 文件的尾部，然后把 *meercat2.txt* 文件的内容添加到 *meercat1.txt*文件的尾部，依次类推，最后把它们合并到一个名为 *meercat.txt*的大文件中。如果你想连续地把输出文件合并在一起（就

像我们前面对 meercats 的操作一样），并且想让它们以文件中的顺序排列，可以输入下面的命令：

```
% cat meercat*.txt > big-meercat.txt
```

该命令将产生相同的结果。像 * 这样的通配符会严格遵照字母的顺序：如果 *meercat10.txt* 和 *meercat11.txt* 文件存在的话，它们会排在 *meercat2.txt* 文件前面。

cat 命令也可以把文件添加到一个现有文件的尾部。例如，如果程序产生了另外一个输出文件，它也需要添加到输出文件集的尾部，则使用下面的命令：

```
% cat meercat10.txt >> big-meercat.txt
```

可以实现该结果。如果在这种情况下使用 > 来代替>>，则不但不会把它添加到集合文件的尾部，而且新文件 *meercat.txt* 将覆盖 *big-meercat.txt* 文件的整个内容。

顺便说一下，如果你需要一个和 *cat* 命令相反的命令，以便以相反的须序打印出文件的每一行，那么正好有这样一个命令：该命令被称为 *tac*。不幸的是，用于彻底反转打印一个文件的 *acta* 命令还没有实现。

more 在正确的方向上前进一步

用法：`more -[options] [+linenumber] [+/pattern] filename`

more 命令是一个翻页器，在 Unix 中意味着这样的一个程序，它可以使你一次浏览一个文件。假设你有一个名为 *blast-first.txt* 的包含 BLAST 输出的文件。输入下面的命令：

```
% more blast-first.txt
```

将显示 *blast-first.txt* 文件的第一页，每次当你按下空格键时，会向前翻一页。如果想离开 *more*，则点击 *q* 键；如果想浏览 *more* 中的更多命令，点击 *h* 键。

在移动文件的时候，使用 *more* 十分智能。如果你知道自己想到文件的哪个位置，可以指定行号（使用 *+linenumber* 选项）。另一方面，如果你想从某个单词或者原型的第次出现时开始，可以使用 *+/pattern* 选项。当浏览更多的文件内容时，如果你按下 / 键，然后输入一个要搜索的原型，就会跳到文件中该原型下一次出现的地方；按下／键后，再按下 Enter 键，可以不断搜索该原型的每一个出现位置。

下面是 *more* 命令的其他一些有用选项：

- *-r* 正常地显示不可打印的控制字符和正常的文本
- *-s* 把多个空行压缩为一个空行

你可以重新定向一个程序的输出，该程序把超过一屏显示的文本分为多个文档，使你一次可以浏览屏的输出信息。如果你想要知道谁登录到你的 Unix 系统上，应怎么办呢？如果登录进了足够多的用户，则显示登录者信息的输出将会出现滚动条。通过把 *who* 管道给 *more*:

```
% who | more
```

可以在输出中使用 Return键一行行地滚动，或者使用空格键一屏屏地滚动。

more 命令的最突出的缺点是，一些版本的 *more* 命令不能在文件中向后滚动信息。*less* 工具弥补了这个缺陷。

less ：黄金标准

这里介绍的是一个高级页面命令 —— *less*。更重要的是，*less* 修正了 *more* 的最大缺陷：它可以使文件中的页面和向前滚动一样，向后也可以滚动。 *less* 在工作的时候，也不是一次把一个文件装载到内存中，如果这样的话，就很可能当你使用 *less* 浏览一个大文件时，计算机处理速度将会逐渐变慢。最后， *less* 命令还可以更加优雅地处理二进制文件，把可读文本显示为字符，以 ^*X* 的形式表示不可读的控制字符。*less* 命令使用和 *more* 相同的选项，但是它也有其他的一些选项。使用 *info less* 命令可以查询在你的本地系统上，*less* 采用的是什么选项。最后要说的是，即然它很少被提及，但是为什么它命名为 *less* 呢？这是因为 *less* 就是 *more*。

使用 vi 和 vim 编辑文件

用法： vim *filename*

因为 Unix 是一个基于文本的操作系统，在历史上曾经用做软件开发和计算，所以 Unix 传统上没有提供在个人计算机上可以实现的“所见即所得（WYSIWYG）”的文本编辑器的全部特征，虽然现在也可以获得这样的编辑器。事实上， WYSIWYG

文本编辑器对程序员来说，其功能有限，因为它们通常会引入不可见的标记字符到文档中。

学习为Unix提供的无格式编辑器十分必要。学习它们的过程是曲折的，但是如果你正在编程或者阅读无格式的文本数据，那么它们正是针对这些工作的正确工具。如果你从一个Web服务器下载了序列数据，将其在一个无格式的文本编辑器中打开并工作，那么你所写出的文件应该可以被序列分析程序读取。如果你在一个WYSIWYG 编辑器中打开相同的文件并工作，那么就会输出该编辑器所使用的格式的文件，这时就无法被其他程序读取。

*vi*编辑器是大多数 Unix 系统的标准特性。它是一个全屏编辑器；它允许你阅读尽可能多的正编辑的文件行，该编辑器将使编辑窗口和终端屏幕或者你运行文件的窗口相匹配。可以使用键盘指令使光标在文件中移动，但是不能使用鼠标移动光标。屏幕底部的一行称为状态行（status line)。来自 *vi*的错误消息显示在状态行中。

在后面的小节“正则表达式的语言”中，我们将讨论使用正则表达式作为无格式文本编辑器 *vi*的一个特性来进行搜索和替换。结合正则表达式语言来使用 *vi*的能力，使得 *vi*成为文件操作的一个强有力的工具。

在 *vim*（增强的 *vi*）中增加了一些新特性。虽然在 *vim*中仅仅添加了多次撤消特性，但是如果你的系统中还没有安装 *vim*的话，还是非常值得请系统管理员来安装 *vim*的。我们在这里不会介绍*vim*的所有特性，但是我们将介绍一些使你入门和运行 *vim*的命令（注 3)。

*vim*有三种模式；在每种模式中，来自键盘的输入以不同的方式进行解释：

命令 *(command)*

这是主要的模式；当你开始工作时，你将自动在命令模式下。击键被解释为 *vim*的短命令，这些命令的大多数由一个或者两个字母组成。通常情况下，可以通过单击（有时是双击） Esc 键来返回到命令模式。

输入（*input*)

通过发布任何需要输入的命令，可以达到这种模式。

注 3:　参见参考文献中关于 *vi*的完整参考。

状态行（*status line*）

这种模式用于发布较长、较复杂的命令。为了切换到命令行模式，只需要在命令模式中输入一个分号（;）即可。分号出现在状态行的左侧，输入的所有字符出现在状态行中。结束命令的输入，并按下 Enter 键后，则执行该命令，并返回到命令模式。

下面是一些最有用的 *vim* 命令模式命令：

h, j, k, l

在文件中逐个字符或者逐行地移动光标。它的顺序像视频游戏预操纵杆："h"是向左移动；"l"是向右移动；"j"是向下移动一行；"k"是向上移动一行。在大多数系统中，当使用 *vim* 时，键盘上的箭头键也可以起移动的作用。

w, b

在文本中逐个词向前（"w"）或向后（"b"）移动光标。单词是由空格来界定的。

), (

在文本中逐句地向前（")"）或向后（"("）移动光标。句子被认为是单词的序列，它是由句子结束字符（. ?!）来终止的。

a, A, i, I, o, O

初始化文本的插入位置。"a"和"A"分别表示在光标之后和当前行的结束位置插入文本。"i"和"I"分别表示在光标前和当前行的开始位置插人文本。"o"和"O"分别用于在当前行的下面或上面插入一个空行，并在这个新的空行中开始插入文本。

x, X

分别用于在光标或光标前面删除文本。如果其后跟一个整数的话，则删除光标后面或光标前面的这个整数数目的字符。

s, S

分别用于替换在光标下的字符或在当前行的字符，其方法是删除光标或当前行的字符，然后初始化替代被删除字符的文本的插入。如果跟一个整数的话，"s"表示使用新文本替换该数目的字符，"S"表示替换该数字所指定的行。

下面是一些最有用的 *vim* 状态行模式命令：

:wq

将改变保存到文件中，然后退出编辑会话。“ :w ”可以单独使用，或者与被写入的文件名一起使用。“ :q! ”使得在退出会话时不保存改变。

r] 后跟一个文件名，插入命名文件的整个文本。

:g/pattern/s//replacement/g

在整个缓存中搜索并使用 *replacement pattern* 。如果结尾处没有使用“ g ”，则在任一行中，只替换第一次出现的 pattern。

:number

把光标移到 number 指定的行。

GNU Emacs 编辑器

*vim*是一个非常灵活的编辑器，可以学习使用它来做任何你需要的文本编辑工作。然而，在 Unix 系统上，进行文本编辑时还有其他选项。这也许就是指 Emacs编辑器。Emacs 是一种由自由软件基金会（FSF）开发的编辑软件。它不仅包括用于 TEX 和 LaTEX 文档、用各种编程语言编出的程序和大纲的特殊模式的文本编辑工具，而且还包括一个文件管理器、邮件和新闻阅读器，并可访问在线文档浏览器 *info*。有些书是专门介绍 Emacs 的（参见参考文献），因此我们这里除了推荐之外，就不多加介绍了。如果你是在 Unix 系统上工作，学习使用 Emacs 是你学习计划中最有用的一项。

使用 strings 命令浏览二进制文件

用法：`strings -[options] filenames`

除了前面我们已经讨论的文本文件之外，还有一些不能作为文本读取的二进制文件。它们在大多数情况下是指程序的输出，或者是指可执行格式的程序本身（和源程序相对应）。由于二进制文件和可执行文件是用机器语言表示的，因此它们不是人类可读的。因为存在这样的语言障碍，毫无疑问可以预测，我们不必白费力气来阅读二进制文件。幸运的是你还有一条途径来弄懂二进制文件，就是和该文件的编程者进行交流。但不幸的是，现在的许多程序，例如商业化的隐藏的 Markov 模型软件或

直接把它们内部的数据结构的表示写到磁盘上的数据采集程序，都使用二进制文件来保存特有的数据文件。那么，就算只有 10%的机会，我们还是要介绍一些技巧来从二进制文件中提取信息，以免使得我们发疯。

第一步是使用前面讲过的 *less* 命令或者使用 *strings* 命令。如果文件的任何部分都在无格式文本中，那么立刻就可以读取它们了。 *strings* 命令把文件中的任何可读文本字符都提取出来，并把它们打印到屏幕上。例如，如果你有一个名为 *badger* 的没有文档记载的二进制文件，想看一下它是否包含任何关于它是做什么用的提示，可试着输入：

```
% strings -n 3 badger | less
```

（ *-n*选项通知 *strings* 在一行中有多少可读的文本字符构成了一个字符串。 *-n* 的默认设置为 4。）把输出管道给 *less*，则输出超过一屏时，使你的页面全部显示它。如果输出的是下面的内容：

```
ATCGTACTGATCGTCGATCGTCGATCATGCA
CGTAGCAGTCGATCATCATCGTACTAGCTAG
ATGCCTGAGCTATACACACTAGTCACGATGC
```

你可以想像 *badger*包含某种数据的二进制编码，它们包括一个核苷酸序列或者一个（不很好的）多序列排列。

od 和二进制数据

用法：od -[*options*] *filenames*

有些情况下，可能不仅仅只识别一个二进制文件就行了。在这种情况下， *od*程序可以提供理解二进制文件内容的第一步。在了解 *od*命令之前，让我们首先快速地了解一下二进制如何才能以更加人性化的形式表示出来。

二进制数据没有使用传统上的十进制（以 10为基数）表示法，而通常用 2的幂作为基数进行描述：或者是 8（以 8为基数）个数字，或者是 16（以 16为基数）个数字。采用 8进制的值通常在其前面加一个 O。例如，十进制表示的 25 用八进制表示就是

031（注4）。另一方面，采用十六进制的值通常在其前面加一个 0x，并且使用字母 A～F描述十进制中的 10～15。十进制表示的 25用 16 进制表示就是 0x19。

如果你想深入研究二进制文件的核心，了解具体进行了什么操作，则应该使用*od* 命令来完成八进制的转储（或十六进制的转储）操作，看你的二进制文件是否可以容易地进行转换。输入下面的命令：

```
% od -c badger | less
```

将为 *badger*创建一个八进制转储，这样你就可以一次翻阅一页文件了。转换后的文件形式如下：

```
0000000  \0  \0  \0 001      \0  \0  \0 006   R   T   D   C   Y   G  \0
0000020  \0  \0 006   R   T   D   C   Y   G  \0  \0  \0  \a  \0  \0  \0
0000040 001   1  \0  \0  \0 003   A   R   G  \0  \0  \0 001  \0  \0  \0
0000060 002   C   A  \0  \0  \0 002   B   Z 270   R   ? 200  \0  \0   @
```

*od*命令的主要选项有：

-c 打印出与字节相对应的文本字符

-x 产生文件的一个十六进制转储

-o 产生文件的一个十六进制转储（默认设置）

-d 产生一个无符号十进制值的转储

除非你是一个认真的程序员，否则你不会喜欢阅读二进制文件。然而，我们只是假定你可能会阅读二进制文件，这时，希望这些标准工具可以帮助你找到问题的答案。

转换和过滤器

过滤器（filter）是一些程序，用来获得输入的数据，并把它转换作为结果输出。它们可以完成那些字处理程序和电子制表软件所能完成的工作，例如提取文件的部分

注 4: 这里有一个老笑话：“为什么程序员会把圣诞节和万圣节搞混呢？因为 OCT 31是DEC 25。”（译注 2）

译注 2: 此处语义双关，OCT 31 既可指 10月31 日万圣节，又可指八进制的 31；而 DEC 25 既可指 12月25 日圣诞节，又可指十进制的 25。而八进制的 31 与十进制的 25是相等的。

内容。转换包括对数据格式的一个简单的操作，或者从数据中选择特定的行或字段。在本小节中，我们将讨论一些较常用的过滤器，它们是 Unix 的一部分。这些过滤器可以从标准输入中读取信息，并写入到标准输出中，还允许把它们合并起来，以完成非常复杂的转换（注 5）。

使用 head提取文件首部

用法： `head -number files`

假定你有一个程序输出了一个长数据文件，该文件有很多连接在一起的不同信息表。撇开为什么有人会写出一个程序来创造如此困难的输出问题不谈，这里有一些命令可以使你操作这样的数据，你需要知道这些命令。*head*就是这样的命令。

默认情况下，*head*把指定的一个文件或几个文件的头 10行发送给标准输出。检查一个文件的首部，是了解在文件中是否有些东西没有使用编辑器打开过，或者对文件进行了完整的 *cat* 操作的一种简便的方法。

使用*–number*标志，*head*可用来从文件的首部选择指定数目的记录。 *head*和 *tail*命令组合使用可以从该文件中提取知道其位置的某行。

使用 tail命令提取文件的尾部

用法：`tail [-f] -number files`

默认情况下，*tail* 命令输出文件的最后 10行，或者输出由 *number* 指定的文件的最后几行。如果使用 *-f*选项，则 *tail* 总是给出文件中最后几行的最新输出。这样，当一个文本输出文件被其他程序写入时，可以使用这种方法来监测加入到该文本输出文件中的是什么。

使用 split和 csplit 分割文件

用法： `split -[options] filename`

用法：`csplit -[options] file criteria`

注 5: 如果需要以标准 Unix过滤器不允许的方式转换数据，请参考第十二章中对 Perl脚本语言的讨论。Perl 是一种非常完整和复杂的语言，可以生成无数种特定过滤器。

*split*命令使你可以把一个现有的文件分割为一些指定大小的较小文件。每个较小的文件都由一个后缀（*aa*，*bb*等）来惟一地命名。*split* 的选项有：

-l lines

把文件分割为 *lines*长度的子文件

-a length

使用 *length*长度的字符来形成后缀

如果你有一个名为 *big-meercat.txt* 的文件，你想把它分割为子文件，这些子文件使用单字母后缀，长度为 100行，并把文件写入到名为*net.**的子文件中，执行上述操作的 *split*命令的格式如下：

```
% split -l 100 -a 1 big-meercat.txt meercat.
```

*csplit*也是把文件分割为子文件，但是比 *split* 更灵活，这是因为，除了可以使用分割的行数和字节数外，它还允许使用判别式（criteria)。下面是 *csplit*的一些选项：

-f prefix

使用指定的文件前缀来形成子文件名

-n length

使用一个指定长度的后缀来形成子文件名；子文件后缀由数字而不是由字母组成

*split*判别式可以用两种方法形成：或者是一个可用作判别式的正则表达式，它可由偏移量修改；或者是可以被指定的几行命令。

在 FASTA 格式中的生物学序列数据库也许包含格式的许多记录，如下所示：

```
>identifying header information
PROTEINORNUCLEICACIDSEQUENCEDATA
```

使用下面的命令，*csplit* 命令可以把这样的一个数据库分割为独立的序列文件：

```
csplit -f dbrecord. -n 6 fastadbfile /^>/
```

该文件被分割为用数字编号的子文件，每一个子文件都包括一个序列。

使用 cut 分割文件元素

用法：*cut -c list filenames*

或者：*cut -f list -d delim -s file*

*cut*命令输出一个输入文件的每一行的被选中部分。文件的一行仅仅是字符串中的任意框架，它以一个特定的分界符结束；分界符是操作系统或者程序可识别的特定的非文本字符。在文件中的行以一个 *EOL*（end-of-line）字符串结束；文件本身是以一个 *EOF*（end-of-file）字符串结束。操作文件时，这些字符通常是不可见的，但是对于处理该文件的程序来讲，它们是很重要的。

例如，假定你有一个名为 *sequence_data* 的文件，它包含下面的字符串：

```
ATC  TAC
ATG  CCC
GAT  TCC
```

下面的命令使用 *cut*来输出文件中每一行的第一个字符：

```
% cut -c 1 sequence_data
A
A
G
```

下面的命令用于输出字段 1 和字段 2 的第 1行：

```
% cut -f 1-2 sequence_data
AAT  TAC
```

每个定义行的部分都可以使用 *-c* 选项由本行中的字符数目选择，或者使用 *-f* 选项由本行中的字段选择。字段是指在一行中由分界符所定义的字符的框架。在文件的文本中，使用的最明显的分界符是简单的空格符，但是也可以使用其他字符。字段和列是不同的，列是通过对输入行中的每个字符进行编号来严格定义的。

*list*参数以字符或以字段为界指定被选择的每行的范围。

以单个字符的形式或两个数字的形式表示的列表由一个 –字符分开。可以用逗号把多个单列或范围分开，以选择它们。第 1个或最后一个数字可以忽略，表示 *cut* 在行的开头开始，或者表示 *cut*在行的结尾处结束。每一行中字符和字段的编号从 1开始。

当使用 *-f*选项时，表示 *cut*将计算字段，而不是计算字符，可以使用 *-d*选项指定一个分界符，而不是指定默认的制表符。使用 *-s* 选项可以使 *cut* 命令忽略那些不包含特定分界符的行。这个选项是很有用的，例如，用于忽略一个表中的首部行。

使用 paste 命令合并文件

用法：`paste -[options] files`

paste 命令使你可以从几个文件中把字段合并到一个较大的文件中。与 *join* 命令不同，*join*命令执行的是对两个文件的数据库类型的合并， *paste*纯粹是对文件的机械合并。仅仅靠每个文件中行的行号，把行合并起来：例如， *file1* 的第 1行加到 *file2* 的第 1 行后面，而不管该行是什么内容。除非使用 *-d*选项另外指定一个分界符，否则被添加的数据将由制表符分开。使用 *-s* 选项和仅有的一个输入文件，*paste* 把输入文件中的所有行合并到一个长行中。

paste 可以通过数据分析应用程序准备待读取的数据文件。如果有相同格式的一组文件，并且已经使用其他过滤器命令从它们的每一行中删除相应的信息，你就可以准备一个允许从每一个文件中绘制出相应的信息的输入文件，而不必独立地读取它们。在前面的例子中，我们使用管道化的命令来从一个复杂的输出文件的表中提取一栏，命令如下所示：

```
% head -167 protein1.pka | tail -100 | cut -c30-39 > protein1.pka.data
```

如果有 8 个类似的输出文件，分别表示蛋白质 1 ~ 8，可以使用相同的方式处理它们，然后把你感兴趣的结果加在一起，放到一个大数据文件中，命令如下所示：

```
% paste protein*.pka.data > allproteins.pka.data
```

在该例子中，每个独立的文件如下所示：

```
3.8
12.0
10.8
4.4
4.0
6.3
7.9
```

每个数字表示蛋白质中的一个氨基酸计算后的pKa值。如果你有几组结果，它们可以有意义地合并到一个表中，*paste*创建了一个简单的由制表符分界的表，如下所示：

```
3.8 3.2 3.6
12.0 12.9 12.5
10.8 10.9 11.0
4.4 4.2 4.5
4.0 3.9 4.2
6.3 6.5 6.2
7.9 7.5 8.0
```

然而，理解数据为何能有意义地合并到一个表中，以及如何使用*paste*命令正确地得到所需要的结果，则是你自己的责任。

使用 join 合并数据文件

用法： join -[*options*] *file1, file2*

基于一个指定的连接字段的内容，*join*命令把两个文件合并到一起，在连接字段中有相同值的两个文件的行假定是一致的。假定文件有一个制表格式，它由字段和一个一致的字段分隔符组成，并且，假定文件将按它们连接字段的增序进行排列。*join*的命令行选项有：

−1 fieldnum
在文件 1 中，使用指定的字段号作为连接字段

−2 fieldnum
在文件 2 中，使用指定的字段作为连接字段

−t character
在整个连接操作中，使用指定的字符作为分界符

−e string
使用指定的字符代替空白输出字段

−a filenum
为指定文件中的每一个不成对的行产生输出；可以指定用于两个输入文件；属于其他输出文件的字段是空的

−v filenum

仅为指定文件中的不成对的行产生输出

−o list

从指定字段的列表中建立输出行，而字段列表的格式为*filenum.filenum*;在列表中的多个项可以由逗号或空白字符分开

*join*是一个很有用的从多个文件建立数据表的命令，并且一系列*join*操作可用来建立一个复杂的文件。在下面的简单例子中，有三个文件：

```
mustelidae.color:
badger black
ermine white
long-tailed tan
otter brown
stoat tan

mustelidae.prey:
ermine mouse
badger mole
stoat vole
otter fish
long-tailed mouse

mustelidae.habitat:
river otter
snowfield ermine
prairie long-tailed
forest badger
plains stoat
```

首先，把 *combine mustelidae.color*和 *mustelidae.grep* 合并起来。两个字段的共同之处是，它们都有动物的名称，而且是每个文件的第一个字段。但是在合并时没有保存*mustelidae.prey*。*join*命令的格式如下：

```
% sort mustelidae.prey | join mustelidae.color - > outfile
```

它将产生如下所示的输出：

```
badger black mole
ermine white mouse
long-tailed tan mouse
otter brown fish
stoat tan vole
```

现在，把结果文件和 *mustelidae.habitat* 合并在一起。如果你想把结果以 *habitat animal prey color* 的形式输出，可以使用下面的命令语法：

```
sort -k2 mustelidae.habitat | join -1 2 -2 1 -o 1.1,2.1,2.3,2.2 - outfile
```

上面的命令在前一步中对标准输入和输出文件进行操作，可产生下面的输出：

```
forest badger mole black
snowfield ermine mouse white
prairie long-tailed mouse tan
river otter fish brown
plains stoat vole tan
```

使用 sort 对文件排序

用法：sort -[*general options*] -*o*[*outfile*] -[*key interpretation options*] -*t*[*char*] -*k*[*keydef*]...[*filenames*]

sort 命令可以对一个文件进行排序，也可以对一组文件进行排序，同时把它们合并到一个文件中，或者检查一个文件的排序情况。在数据处理中，这个功能有许多应用。默认情况下，文件中的每一行都被当作是一个单独的字段，但是用户也可以在命令行上定义键。

sort 命令的主要选项有：

-c 根据用户选择的选项来检测一个文件的排序

-m 合并几个输出文件

-u 当几行被认为相等时，只显示其中的一个实例

-o outfile

把输出发送到一个文件中，而不是发送到标准输出中

t char

使用指定的字符来对字段分界

确定如何说明键的选项可以用作全局选项，但是它们也可以用作一个特殊键的标志。*sort* 的键说明选项有以下几个：

-b 在排序键中，忽略头部或尾部的空白区域。

-r 对于某个特殊键而言，颠倒排列顺序。

-d 在进行比较时，使用“字典顺序”；例如，除字母、数字和空格外，忽略其他字符。

-f 为了进行比较，重新把小写字母当成大写字母进行排序。正常情况下，L和 l 之所以被分别对待，是因为它们分别属于大写和小写字符集，而使用 *-f*标志时，则不管是否为大写，所有以 L 结尾的都为大写。

指定排序键

键的定义是指 *-k* 选项的参数。键定义的形式是 *position1,position2*。它们每一个都是一个数值，指定在一行中键的起止位置。位置可以采用*field.character*的形式，这里的 *field*指定在输入行中的字段位置，而 *character*指定在键的独立字段中，它的开始字符的位置。如果键使用键说明选项中的一个标志，键的形式则为 *field.character[flags]*。如果没有把键说明用于整个排序，而只是用于一个键，则把它添加到键定义中，并且没有连字符。

文件统计和比较

在很多的情况下，区别两个文件是否相同（如果不同，其区别是什么）是非常有用的。例如，如果你已经在自己的计算机上编译了一个程序，并且已经提供了测试实例，你就应该在该测试实例上运行程序的副本，并把测试输出和程序编制者提供的标准输出进行比较。如果想检查文件的备份副本和当前的文件版本是否相同，文件比较工具是很有用的。Unix提供了这样的工具，使你不必通过手工繁琐地搜索所有文件。

使用 cmp 和 diff 比较文件

用法：`cmp -[`*options*`]` *file1 file2*

用法：`diff -[`*options*`]` *file1 file2*

假定有两个列表，它们看上去很相似，但你不能凭眼睛看出它们是否真是相同的列

表。如果使用两个略微不同的序列向后进行数据库搜索，以得到一个基因名称列表，并想知道它们是否相同时，就会出现这种情况。为了严格地比较它们（并且在该过程中保护你的眼睛),可以试着使用 *cmp*和 *diff*命令。简而言之，*cmp*命令可用来判断两个文件是否相同，而 *diff* 命令用来打印任何不同的行。

cmp 非常简单，只需键入以下命令：

```
% cmp enolase1.list enolase2.list
```

则在两个文件相同时，不会有输出，否则， *cmp*将返回一条消息，指出文件不同，并且指出第一处出现不同的字符和行。

*diff*对于比较一个文件的不同版本，以确定它们在何处不同时最有作用。在了解 *diff* 的相对繁琐的输出之前，让我们看一下如何来理解其输出。在不用选项时， *diff* 将给出文件间不同之处的列表，其形式是从*file1* 转变到*file2*所需要的改变，如下所示：

x,y d i

在*file1* 中，从 *x*行到 *y*行的数据没有出现在 *file2*的 *i*行之后的数据中（也就是说，它们已经从 *file2*中删除)。

i a x,y

在文件 *file2* 中，从*x*行到*y*行的数据没有出现在文件 *file1* 的 *i*行后（也就是说，它们是后来添加到*file2* 中的)。

i,j c x,y

file1 中的 *i*行到 *j* 行已经改变为 *file2* 的 *x*行到 *y* 行。

在实际应用中，其输出如下所示（这里 *enolase1.txt* 和 *enolase2.txt* 是假定的烯醇酶的名称列表，它们是通过在不同的时候执行的两个数据库搜索产生的）：

```
% diff enolase1.list enolase2.list
1a2
> ENO_MESCR
5a7
> ENOA MOUSE
```

*diff*有两个有直接作用的选项：

–b 忽略被检测行之间的空白区域的不同

−B 忽略在文件中插入或删除的空白行

*diff*及其变体的 *info* 页特别有用。如果需要经常使用 *diff*命令，建议阅读一下这些内容。

使用 wc 计算单词数

用法：wc -[*options*] *filename*(s)

wc 是一个简单、有用的工具，用于计算文本文件中的内容。对于一个给定的文本文件，*wc* 计算该文件所包含的行数、单词数和字节（字符）数。 *wc*命令的默认设置是计算这三种内容，因此，如果在命令行中输入该命令会显示以下结果：

```
% wc meercat1.txt
   27    98    559 meercat1.txt
```

该输出告诉我们，*meercat1.txt* 文件中有 27 行、98 个单词，其大小为 559 字节。如果给 *wc*命令传递了多个文件，它将返回每个文件的信息，以及它们合并起来后的信息。例如，如果你在三个 meercat 文件上运行 *wc* 命令：

```
% wc meercat1.txt meercat2.txt meercat3.txt
```

（或者，为了节约时间，使用有通配符的 *wc meercat*.txt* 命令）其输出如下：

```
     41    130    905 meercat1.txt
     50    124    869 meercat2.txt
     10     19    156 meercat3.txt
    101    273   1930 total
```

下面是 *wc* 命令的一些选项：

−c 仅计算字节（字符）

−w 仅计算单词

−l 仅计算行

--help

打印出命令的使用帮助信息

--version

打印出所使用的 *wc* 的版本信息

通常可以把 Unix 工具结合起来，搜集你所需要的信息。例如，如果有 1000个文件需要处理，并且输出文件都保存在相同的目录中。可不使用 *ls* 命令来列举那个目录的内容，而是使用 *ls -1 dirname | wc* 来浏览到现在为止已经创建了多少个输出文件。

正则表达式的语言

像正则表达式（regular expression）这样的模式匹配语言可用来在文件中搜索和提取匹配的部分，并且可用来替换与模式字符匹配的字符（在使用正确程序的情况下）。正则表达式用在 *vi* 和 Emacs 文本编辑程序中。因为生物学家所处理的数据中大多包含模式字符，所以我们需要学习的第一个技巧就是如何匹配模式，并把它们从文件中提取出来。

正则表达式也可以被 Perl语言解释器识别。了解了如何使用正则表达式，并了解了 Perl 的基本命令，我们就会拥有一个强大的数据处理工具集。我们将在这里介绍正则表达式的基本概念，在第十二章中还会继续介绍它。

如果你曾经在搜索时使用了通配字符，就说明你已经使用了一个正则表达式。正则表达式是指要匹配的文本的模式。在正则表达式中也可以使用专用的字符代表变量模式，这意味着可以进行局部的或不精确的匹配。正则表达式可以由明确的文本和特定字符的任意结合构成。

在基本的正则表达式中，可识别的专用字符如下：

\ 在反斜线的后面有一个特定字符时，反斜线的作用是该字符的转义字符。如果你正搜索的查找内容中有点号，例如 *chars.txt*，可以给出正则表达式 *chars\.txt* 来搜索它。

点号匹配任意单个的字符。

在正则表达式中，星号的作用与它作为 shell 通配符时的作用不同。如果用在一个字符的前面，它用于匹配 0，或者在那个字符再次出现时匹配它。如果用

在一个字符类描述前面，它用于匹配 0，或者在那个集合中再次出现该字符类描述时匹配它。如果用在一个点号前面，它用于匹配 0或者更多的随意字符，这时，它的作用和在 shell 中所起的作用是相同的。

如果在一个正则表达式的开始有 ^ 符号，表示它匹配一行的开始。否则，它匹配它自己。

$　如果在一个正则表达式的尾部有 $符号，表示它匹配一行的尾部。否则，它匹配它自己。

[charset]

在中括号中包围的一组字符可以匹配括号中的任意单个字符。例如，*[badger]* 可以匹配（a, b, d, e, g, r）中的任意一个字符。在该集合中，只有 -、^、]和\ 是不可匹配的。其他的所有字符，包括常规的特殊字符，都可以匹配其自己。也可以使用 [*c1*-*c2*]形式的字符范围；例如，[0-9] 或[A-Z]。

使用 grep 搜索模式

用法：grep -[*options*] '*pattern*' *filenames*

grep 允许在一个文件或一组文件中搜索模式（以正则表达式形式）。根据所选择的下面三个功能中的一个，GNU *grep*（在 Unix 上是标准命令）可以搜索三类模式：

-*G* 标准 *grep*：搜索一个正则表达式（为默认选项）。

-*E* 扩展 *grep*：搜索一个扩展正则表达式。

-*F* 快速 *grep*：快速地搜索一个固定的字符串（与正则表达式相对应，该字符串是一个由正常字符组成的模式）。

注意，-*E* 和 -*F* 选项在一些系统上可以通过调用 *egrep* 或 *fgrep* 来显式选择，如果没有指定要搜索的文件，*grep*搜索该模式的标准输入，如果正在输出中搜索一个模式，则允许其他程序的输出重定向到 *grep*。

下面的命令是一个简单的例子：

```
% grep -c '>' SP-caspases-A.fasta SP-caspases-B.fasta
% grep '>' SP-caspases-A.fasta SP-caspases-B.fasta
```

上面的两个命令都在一个 FASTA 格式的序列文件中搜索（应该记住，该文件的头几行以>符号开始）。第一行命令返回每个文件的序列数目，而第二行命令返回一列序列头。但是，不要忘记用>符号结束。否则，一旦发现有此问题，命令将被解释为使用 *grep* 的请求来搜索一个没有模式的标准输入，然后把结果的空字符串重定向到所列的文件，不管有什么文件，都会覆盖它。

grep 有大量的选项。下面是一些比较有用的选项：

–c 只打印几个相匹配的行，而不打印匹配行自身。

–i 在文件和模式中忽略大写／小写。

–n 在出现与模式相匹配的行时，打印该行和行号。

–l 打印含有和模式相匹配的文件名，但不打印相匹配的行。

–h 打印相匹配的行，但不打印文件名（和 *–l* 选项正相反）。

–v 仅打印不包含和模式相匹配的那些行。

–q （quiet 模式）在第一次出现之后，停止列出匹配。

在蛋白质结构文件中，蛋白质序列信息以三字母代码序列的形式保存，而不是以更为紧凑的单字母代码格式保存。因此，有时需要从蛋白质结构文件中提取序列信息。在实际情况中，可以使用一个简单的 Perl 程序来完成该工作，然后继续把序列翻译为单字母代码。但是也可以使用两个简单的 Unix 过滤器命令来提取序列。

第一步是在 PDB 文件中找到 SEQRES 记录。这一步可以使用下面的 *grep* 命令来进行：

```
% grep SEQRES pdbfile > seqres
```

完成上面的命令后，将给出一个名为 *seqres* 的文件，它包含如下所示的记录：

```
SEQRES 1 357 GLU VAL LEU ILE THR GLY LEU ARG THR ARG ALA VAL ASN 2MNR 106
SEQRES 2 357 VAL PRO LEU ALA TYR PRO VAL HIS THR ALA VAL GLY THR 2MNR 107
SEQRES 3 357 VAL GLY THR ALA PRO LEU VAL LEU ILE ASP LEU ALA THR 2MNR 108
```

由于在每个记录中，并不是所有的字符都属于氨基酸序列。因此，下一步需要从记录中提取出序列。可以使用 *cut* 命令来完成该工作：

```
% cut -c20-70 seqres > seq
```

该命令的输出保存在 *seqs* 文件中，格式如下：

```
GLU VAL LEU ILE THR GLY LEU ARG THR ARG ALA VAL ASN
VAL PRO LEU ALA TYR PRO VAL HIS THR ALA VAL GLY THR
VAL GLY THR ALA PRO LEU VAL LEU ILE ASP LEU ALA THR
```

如果不想创建中间文件，可以把上面的两个命令管道到一个命令行中，如下所示：

```
% grep SEQRES pdbfile | cut -c20-70 | paste -s > seqs.
```

paste –s 命令的其他功能是把文件中的独立的行连接到一个长行中。

Unix Shell 脚本

各种 Unix shell提供了编写多步脚本的机制，这样就可允许你自动进行工作。之所以称为脚本(script)，是因为它们逐字地包含你想要提供给 shell 的命令序列，这就像在演戏过程中，编剧要求演员所说的台词剧本一样。

即使是最简单的脚本，shell 脚本，也是应用程序，因此，它们可以进行工作。假定你想开始一系列计算，这需要一会儿，然后你回家吃饭。默认情况下， shell 将等待一个命令结束后才执行下一条命令，因此第二条命令将按照第一条命令的输出进行工作，它不会提前执行。重要的是，不需要等在那里输入第二条命令。

下面有一个相对简单的例子。假定你已经下载了整套 GenBank DNA序列文件。你想得到这些文件中的信息，但是你需要这些信息以与原来不同的格式表示，这样你下载的程序才能处理它。你计划使用 *gb2fasta* 程序把 GenBank 中的文件转换为 FASTA 格式。（该脚本假定已经把 GenBank 文件下载到了你当前的工作目录中。）然后你想使用 BLAST *formatdb* 程序处理每个文件。为了使脚本更加灵活，你可以编写下面的脚本，这样就可以在命令行上有一个可选择文件列表，来指定哪个文件需要进行处理。该脚本如下所示：

```
#!/usr/bin/csh
foreach file ($*)
echo $file
gb2fasta $file > $file.na
formatdb -t "$file" -i $file.na -p F
end
```

在创建文件之后，你需要使用 *chmod* 命令使其成为可执行文件。例如，如果脚本的文件名为 *blastprep*，则可以使用下面的命令：

```
chmod a+x blastprep
```

脚本的第 1 行将通知操作系统使用的是哪个 shell 程序，shell 是否被调用，以及任务是否正在运行。可以使用下面的方式直接调用命令：

```
./blastprep gbest*.seq
```

为了不使用全路径就可以运行新的脚本，需要在键入运行脚本的命令前，先运行 *rehash* 命令。*rehash* 是一个 C shell 命令，它可以更新路径上的所有可执行文件的列表。

在前面的例子中，所有 GenBank EST 文件都会自动进行语法检查，以准备用于 BLAST。*gb2fasta* 和 *formatdb* 程序的运行就像它们在命令行中一样，但是你不必等待每一行结束。脚本在 *gbest*.seq* 中（这是一个文件名列表）使用命令行参数，并且使用每个值依次填充变量 *$file*。然后，脚本在“ foreach ”行和‘ end ”行之间循环。*echo* 命令仅仅是把 *$file* 的值发送给标准输出，因此你可以在终端窗口中看到任务是如何执行的。正常情况下，*gb2fasta* 程序打印到标准输出，因此需要把输出重定向到一个特殊的文件名。另一方面，*formatdb* 处理输入文件，并使用内部命名惯例生成新文件，所以在脚本中不需要输出文件。

和其他计算机进行通信

正像我们将要在第六章中看到的那样，由于具备与世界上其他计算机和网络相联的能力，我们可以读取和下载大量的信息，并且可以和同事共享数据。事实上，生物信息学家的主要的工作是和公共数据库及其他生物数据仓库打交道。在本节中，我们将介绍计算机是如何和其他计算机通信的，以及在通信过程中使用的是什么工具。

Web

和其他计算机进行通信的最简单方法就是通过 Web。大多数 Linux 的发布版本都提供了 Web 浏览器，通常是 Netscape，如果你在安装的时候选择该安装选项，它将会

自动安装到你的计算机上。在 Linux 系统上安装 Web 浏览器的过程和在其他计算机上安装浏览器的过程一样；需要进行浏览器的参数设置，并且告诉它打开不同种类文件附件的工具位于什么地方。

你也许想在自己的机器上维护一个主页，为了达到这个目的，需要安装 Web 服务器软件。大多数 Linux 发布版本也允许在安装过程中选择 Apache Web 服务器软件作为其中的一个安装选项。如果选择安装了一个 Apache Web 服务器，你就可以把适当的 HTML 文件放在 */home/httpd/html* 目录下，以发布一个简单的网站。

IP 地址和主机名

在因特网的世界中，计算机是通过它们的因特网协议（IP）地址相互识别的。那些一直连接在因特网上的计算机都分配了永久性的 IP 地址和主机名；而那些只是偶尔连接到因特网的计算机则根据它所使用的连接协议只有动态分配的 IP 地址，或者没有任何 IP 地址。

IP 地址由用点号分开的四个数字组成（例如，128.174.55.33）。它们被解释为通过网络软件找到主机（host，指一个和其他计算机通信的计算机）的说明。计算机也有主机名（hostname），例如 *gibas.biotech.vt.edu*。名称服务器（name server）是专用的机器，它维护着 IP 地址和主机名之间关系的信息。

telnet

用法：`telnet` *`full.hostname`*

telnet 命令用于在远程 Unix 机器上打开一个 shell；发布命令的工作站成为该机器的一个终端。为了 *telnet* 到其他的 Unix 机器，必须首先登录到该机器上。一旦登录到远程主机上，shell 将像直接在远程机器上工作一样进行工作（注 6）。

在输入你的 ID 之后，接着出现一个“password:”提示符，然后出现一个“login:”提示符。

注 6：　如果以 *root* 身份登录，有些任务就不能从远程终端完成。

ftp

用法：`ftp full.host.name.edu`

ftp（File Transfer Protocol，文件传送协议）是一种把文件从一台计算机传送给另一台计算机的方法。你也许熟悉了 Fetch、Interarchy 或者其他基于 PC 的 FTP 客户端；Unix *ftp* 在概念上和这些程序类似（这些程序在 Linux 环境下大多数有它们的类似物，这些类似物具有图形用户界面）。使用 *ftp* 连接到其他主机上时，你将发现自己是处于一个独特的 *ftp* 操作环境中。虽然 *ls* 和 *cd* 命令有类似的功能，但是 Unix 命令并不总是工作在 *ftp* 环境中。

在“password:”提示符后面，会再次出现“login:”提示符。如果你正在访问一个匿名 FTP 服务器（发布软件的一种通用方式），则标准的用户名是 *anonymous*，并且你的 email 地址就是密码。一旦进入 FTP 环境中，你应该知道的最重要的命令如下所示：

help
: 打印 *ftp* 命令的列表。*help* 命令打印出一个特定命令的信息。

ls
: 在远程主机上列出目录的内容。

cd
: 在远程主机上改变工作目录。

lcd
: 在本地主机上改变工作目录。

get, *mget*
: *get* 命令把单个文件从远程主机复制到本地主机。*mget* 命令则用于复制多个文件。

put, *mput*
: *put* 命令把单个文件从本地主机复制到远程主机。*mput* 命令则用于复制多个文件。

binary，*ascii*

把文件传送模式由二进制模式修改为 ASCII模式。下载二进制可执行文件、图像和其他编码的文件格式时，应该选择二进制模式。

prompt

在传送多个文件时，轮换交互模式，这样在每次传送时都会要求你进行确认。

从远程终端中显示

有时需要在其他计算机上运行 X 程序，并且把它显示在你的终端上。这种工作相对来说比较简单。首先，你需要设置自己的终端允许来自其他主机的远程显示。可以使用 *xhost* 命令来完成这个工作，如下所示：

```
xhost +
```

在确认可以从其他主机进行访问之后，把该确认打印到标准输出。

下一步，需要修改在远程机器上的显示环境。可以使用 *setenv* 命令来完成，如下所示：

```
setenv DISPLAY yourmachine.yoursubnet.wherever.edu:0
```

并不是运行在远程服务器上的所有 X 应用程序都可以使用你的终端来显示，这通常是由于远程机器和你的机器没有相同的图形功能。例如，在一个远程 SGI 机器上运行的程序不能在你的本地 Linux工作站上显示，这是因为当前的 Linux 用户并没有 SGI 所使用的相应的图形库。然而，即使两个机器是兼容的，带宽限制也会使得在网络上运行大型 X 程序非常慢。

通信和文件共享

对于 Linux用户来讲，当他们处于 Mac/PC 环境中时，一个最不方便的地方在于如何和其他用户共享由 PC 软件所产生的文件。在这里，虽然我们的目的不是教你使用这些软件包，但是我们会提到一些选项，以帮助你处理如何和非 Unix 用户进行通信。

幸运的是，在 Linux 环境下有一些相对费用比较低的软件产品，使它可以和常见的

```
Detected scsi disk sda at scsi0, channel 0, id 0, lun     SCSI device
sda: hdwr sector= 512 bytes. Sectors= 35566499 [17366   MB] [17.4 GB]
sda: sda1 sda2 sda3 sda4 < sda5 sda6 sda7 sda8 sda9 >
```

像 Unix 文件系统那样访问设备

一旦知道了设备ID，那么加载这些新文件系统就很简单了。如果你是自己机器的根用户，使用的命令如下所示：

```
mount -t [filesystem type] devicefile mount point
```

例如，如果想加载位于 */mnt/floppy* 的 PC 格式的软盘驱动器，使用的命令如下:

```
% mount -t msdos /dev/fd0 /mnt/floppy
```

可以在 *mount* 手册页中找到允许的文件类型列表。

作为简写形式，可以把自己的 */etc/fstab* 文件修改为包含下面的几行:

```
/dev/fd0        /mnt/floppy       vfat   noauto,owner   0 0
/dev/hdd4       /mnt/zip          vfat   noauto,owner   0 0
```

在这个系统上，Zip驱动器位于 */dev/hdd* 中。所有 PC格式的 Zip 盘都使用分区数 4，并且用于该分区的设备文件是*/dev/hdd4*。*noauto*标志表示这些磁盘不能在启动的时候自动地加载。一旦这些行被添加到 */etc/fstab* 中，就可以使用简写命令 *mount devicefilename* 来加载这些设备。

一旦 Zip或者软盘作为一个分区加载，在那个磁盘上的文件就可以当作是系统上的任何其他文件一样进行处理。

使这些设备工作就不像我们希望的那样简单了。为了获得进一步的帮助，可以搜索关于 Linux 知识的页面，学习如何使用这些特定的设备。

作为 DOS盘的访问设备

如果安装了应用软件包*mtools*和它的图形前端软件包 *mfm*，你可以运行 *mfm*，使用类似PC 上的图形界面把文件移动到 Zip盘或者软盘上。但是，如果使用这个方法来访问设备，那么在把保存在媒体上的文件移动到本地硬盘之前，不能在它们上运行 Unix 命令。

默认情况下，访问媒体的进程也许仅仅由根用户运行。如果对你的系统进行配置，那么其他的用户也就可以写入到软盘驱动器和 Zip 驱动器了。但是，这就会在你的系统上产生一个安全隐患。你必须保证进行方便的磁盘访问的好处要大于任何潜在的危险造成的损害。

在共享环境中和其他人轻松交流

传统意义的 Unix 环境一直是多用户环境。而用于个人计算机的 Unix 可能会有所改变，因为你在家里或者工作中可能会经常使用一个共享的、网络化的 Unix 系统。但是，即使是在个人 Unix 系统中，在自己的系统上创建一个过载时，你也需要注意可能会出现的问题；注意后台进程如何干涉你运行交互式进程的能力。

因为Unix操作系统一次可以和多于一个的用户进行交互，这些用户可以在直接和系统相连的终端上，或者在网络上的某个位置，因此在你的系统上可以执行很多进程。一些进程是你的进程，其他的则可能属于某个正在几百英里外的房间中工作的用户。为了成为 Unix环境中的一个好公民，你需要共享系统的资源。虽然大型公共系统的管理员通过对空间使用实现限额，以及通过进行系统排队以便进行进程管理使你几乎不可能成为一个坏公民，但不可能所有的系统都会进行如此严格的管理。在共享系统上，一个好的规则就是每个用户都不要越雷池一步，尽责地管理好自己的进程是明智的。否则别人就会和你发生冲突，这样就不妙了。

进程和进程管理

Unix 系统在同一时间内执行许多不同的操作。每一个操作或者进程都有一个独一无二的 ID，它允许用户或者管理员直接和那个进程进行交互。

不管在初始时是否激活进程，在你的系统上总是会运行最小数量的进程。每个 shell 程序，不管它是空闲还是激活，都有一个进程 ID 和它相连。几个系统（或者根）进程，比较有名的是 daemon（后台程序），一直在系统上处于激活状态。这些进程通常会等待你初始化一些它们处理的活动：例如，打印文件、发送 email 或者初始化一个 *telnet* 会话。

在最小的系统活动层次外是你初始化的一些进程，它们是通过输入命令或者运行程序初始化的。Unix 内核管理这些进程，不管什么资源可以用于这些进程，可以根据它们的需要进行分配。

每个进程都使用系统的一个 CPU 或者几个 CPU 处理能力的百分之几。它也使用系统内存的百分之几。当运行在一个机器上的进程需要超过CPU处理能力的 100%时，每个单独的进程都会执行得更慢。虽然 Unix 可出色地在同一时间内运行上百个进程而不死机，但最后当机器上的负载增加到机器所无法承受的水平时，机器仍会变至没用。操作系统采用了很多技术来避免这种情况的发生，例如限制进程可以开始的绝对数目，以及交换溢出内存的空闲任务。甚至在单个处理器系统下，只要在内存中有把两个任务都放进去的足够空间，也可能会同时运行多个进程。在CPU 一直等待数据从硬盘上的交换空间中装载的时候，你会看到一个低效率期。这一点可以使用 *top* 命令来监视，我们将在“ top ”部分介绍它。相对使用低频率的 CPU 而言，很多机器更多地受限于内存的缺乏，因此，现在更加有效的方法是花钱买 RAM 来添加内存,而不是购买最新、最好和最快的 CPU。

检查负载的平均水平

用法： w

在大多数 Unix 系统上都可以使用 *w* 命令。该命令可以显示哪些用户已登录到系统上，以及他们正在做什么。它也可以显示系统的当前负载平均水平。

w 命令的标准输出如下所示：

```
2:55pm up 37 days, 4:50, 4 users, load average: 1.00, 1.02, 2.00
USER     TTY    FROM        LOGIN@    IDLE    JCPU   PCPU  WHAT
jambeck  tty1              22Jan99  37days   3:55m  0.06s  startx
jambeck  ttyp0  :0.0       Wed 5pm   1:34m   0.22s  0.22s  -csh
jambeck  ttyp3  :0.0       21Feb99    3:47   9.05s  8.51s  telnet weasel
god      ttyp2  around      2:52pm   0.00s   0.55s  0.09s  create world
```

输出的第 1 行是首部。显示了日期，机器已经运行了几天，有多少用户登录进来过，以及在最后 1分钟、5 分钟和 15分钟系统上的负载平均数为多少等内容。负载平均数表示在系统上所使用的处理器的百分比。如果你使用的是一个单处理器系统，其负载平均数为 1，则该系统正在优化状态下使用。如果你使用的是一个具有四个处理器的系统，其负载平均数为 2，则表示处理器在其一半能力下使用。如果你登录

上一个系统，它已经在其能力下使用，或者超出了其能力，应该添加其他处理器，这样它们立刻就可以运行。使用 *batch* 或 *at* 命令可以安装一个处理器，当资源可用的时候，可以使进程开始。

显示给每个用户的信息是用户名、tty名、用户登录进的远程主机、登录时间、空闲时间、JCPU 和 PCPU时间，以及用户正在做什么。

使用 ps 列出进程

用法: ps [*options*]

在发布命令时，*ps* 命令生成处理器正在处理任务的快照。根据计算机当时正在处理什么，在提示符后输入 *ps* 命令将产生下面的输出：

```
  PID TTY     TIME CMD
36758 ttyq10 0:02 tcsh
43472 ttyq10 0:00 ps
42948 ttyq10 4:24 xemacs-20
42967 ttyq10 1:21 fermats-last-theorem-solver
```

大多数 *ps* 的选项修改了 *ps* 报告进程的类型，并且修改了 *ps* 报告进程的方式。下面是一些较有用的选项：

a 列出计算机上运行的所有命令，包括其他用户使用的命令

l 产生一个进程的长列表（处理内存的大小、用户 ID 等等）

f 以“树”状形式列出进程，显示相关的进程

注意，你不必急于使用这些选项。实际上 *ps* 还有很多选项；使用 *info ps* 可以了解在你本地安装的机器上都支持哪些选项。

top

用法: top -[*options*]

top 命令提供了对处理器活动的实时监控。它列举出系统上由 CPU 使用、内存使用或运行时所排列的进程。上面的屏幕显示如下：

```
4:34pm up 37 days, 6:29, 4 users, load average: 0.25, 0.07, 0.02
42 processes: 39 sleeping, 3 running, 0 zombie, 0 stopped
CPU states: 42.9% user, 6.4% system, 0.0% nice, 51.0% idle
Mem:  39092K av, 38332K used,   760K free, 13568K shrd,  212K buff
Swap: 33228K av, 20236K used, 12992K free              8008K cached

  PID USER     PRI NI  SIZE  RSS SHARE STAT LIB %CPU %MEM   TIME COMMAND
  516 jambeck   15  0  4820 3884  1544 R      0 30.4  9.9   4:23 emacs-fgyell
  415 root       9  0 10256 9340   888 R      0 15.5 23.8 161:41 /usr/X11R6/b
10756 cgibas     5  0   716  716   556 R      0  2.3  1.8   0:01 top-ci
```

首部类似于 *w* 的输出，但是信息更详细些。它给出了 CPU 和内存使用的详细分目，包括计算机的正常工作时间和负载平均数。该显示可以改变，用于显示各种字段。*top* 的默认配置是在用户的 *.toprc* 文件或系统下的 */etc/toprc* 文件中设置的。

下面是 *top* 的一些选项：

-d 使用延迟（delay）的频率进行更新

-q 没有任何延迟即进行刷新，在最高优先权下运行

-s 禁用最有潜在危险的命令，以安全模式运行

-c 打印整个命令行，而不是仅仅打印正在运行的命令

-i 除了当前正在运行的那些进程外，忽略其他所有的进程

当 *top* 运行的时候，可以输入某些交互命令，除非这些命令在命令行上禁用。*i* 命令可以在显示所有的进程和仅显示当前运行的进程之间进行转换。*k* 用于结束进程。该命令提示你将要结束进程的进程 ID 以及发送给该进程的信号。信号 15 是指正常的结束；信号 9 是指迅速、彻底地结束进程，它不会被进程忽略。*r* 改变进程的运行优先级，实现了将要在后面讨论的 *renice* 命令。它提示你输入进程 ID，以及该任务的新的优先值。

使用 kill 给进程发信号

用法：`kill [-s` *`signal`* `| -p] [-a]` *`PID`*

kill 命令允许在命令行中不正常地终止一个进程。虽然 *kill* 实际上可以发送各种类型的信号给一个进程，但是，实际上它最常用的形式为 *kill PID*；或者，如果这样不能结束该进程，则使用 *kill -9 PID*。

在大多数系统上，*kill -l* 可列出各种可以发送给一个进程的信号。有时，了解下面的知识很有用：任务可以使用 *kill -s STOP* 和 *kill -s CONT* 命令来停止和重新开始（注 7）。

PID 通常仅仅是数字表示的进程 ID，可以使用 *ps* 或者 *top* 命令来显示它。PID也可以是一个进程名，在这种情况下，可以寻址一组类似命名的进程。PID 的另外一种有用的形式是 *-n process group ID*，它允许 *kill* 命令同时在一组进程中寻址所有的进程。

使用 nice 和 renice 命令设置进程的优先级

用法：nice -n [*val command arg*]

用法：renice -n [*incr*] [-g|-p|-u] *id*

除非你让 Unix 系统初始化时的进程做其他事情，否则它们具有最高的优先级。*nice* 和 *renice* 命令允许进程的使用者或者超级用户来降低一个任务的优先级。

如果有限的计算机资源由许多用户共享，并且计算机同时用于计算和交互工作，比较好的策略是使用低优先级运行后台任务（指那些在计算机上运行，但是没有任何交互界面的任务）。否则，当后台任务占用可用的资源时，像文字编辑或者图形显示程序这样的交互任务会运行得非常慢。当较高优先级进程运行的时候，在低优先级下运行的任务将减慢。当系统上的负载下降时，低优先级的后台任务就会扩张，使用所有的可用资源。

可以使用 *nice* 命令在低优先级下初始化一个命令。*n* 是优先级值。在大多数系统上，默认情况下 *n* 设置为 10，其范围可以从 1 ~ 19 或者 0 ~ 20。当然，数字越大，任务的优先级越低。

renice 命令允许重新设置一个已经运行的进程的优先级。*incr* 是添加到当前优先级的一个值。因此，如果你有一个以正常优先级（优先级 1）运行的后台进程，并且你想（通过增加其优先级的值）降低它的优先级，可以输入 *renice -n 18* 命令来把优先级值增加到 19。也可以给在高优先级下运行的任务输入一个负数，但是除非你

注 7：有关其他信号的讨论可在参考文献列出的综合 Unix 参考书中找到。

是根用户，否则不能把它的优先级提高到 1。*renice* 的选项 *-p*、*-g* 和 *-u* 使 *renice* 分别把 *id* 解释为进程 ID、进程组 ID或者用户数目。

使用 cron 安排循环活动

*cron*的后台程序 *crond* 是一个标准的Unix 进程，它执行系统和单个用户的连续任务。像清空 */tmp* 目录和系统备份这样的系统活动就是由 *cron*后台程序控制的典型功能。假定普通的用户被授权可以运行*cron* 任务，那么他们也可以把自己的任务提交给*cron*。关于 *cron*授权的细节可以参考*crontab* 手册页。由于后面将讨论的*at*和*batch* 命令也由 *cron*控制，所以默认情况下，大多数系统都配置为允许用户使用 *cron*。

使用 crontab 把任务提交给 cron

用法: crontab -[*options*] *file*

提交给 *cron* 的任务由 *crontab* 命令来完成。*crontab -l > file*命令把 *crontab* 的当前内容放到一个文件中，这样你就可以编辑这个列表。 *crontab file* 发回最新编辑过的文件，并且为了 *cron* 命令能使用而初始化它。 *crontab -r* 删除当前的 *crontab*。

cron 处理所有 *crontab* 的内容，然后按计划初始化任务。一个由 *crontab -l* 产生的 *crontab* 入口如下所示:

```
# Format of lines:
#min  hour  daymo  month  daywk  cmd
  50    2     *      *      *   /home/jambeck/runme
```

该入口在每天早上 2:50运行程序 *runme*。在任意字段的星号表示“每次执行这个功能”。在这个入口中,所有向 STDOUT 或 STDERR 的输出都发送电子邮件到用户 *jambeck* 的邮件账号，其账号在 *cron* 任务运行的机器上。

使用 cron 安排一个循环的数据库搜索

究竟为什么你的组成员以每天为基础执行DNA排序呢？为什么你想使用序列排序程序 BLAST 自动地把你的序列和一个非冗余蛋白质数据库相比较呢？考虑一下下面的这个 *crontab* 入口：

```
01 4 * * * find /data/seq/ -name "*.seq" -type f -mtime -1 -exec
/usr/bin/csh /usr/local/bin/blastall -p blastx -d nr -i '{' ";'
```

这个入口自动在 4:01 A.M.运行，并且检查在最近24小时中数据库内被修改或添加的所有序列。然后，它运行 BLASTX 程序以搜索非冗余蛋白质序列数据库的副本，以匹配新序列，并把结果用电子邮件发送给你。这个例子假定你正确地进行了所有必要的环境变量设置，这样 BLAST 就可以找到必要的记分矩阵和数据库。它也使用一个默认的参数设置，该参数设置可能需要修改以得到有用的结果。一旦正确地配置了它，你所要做的就是在每天早上喝咖啡的时候浏览一下邮件。

使用 batch 和 at 命令安排进程

用法：`at -[`*`options`*`]` *`time`*

用法：`batch -[`*`options`*`]`

batch 和 *at* 命令是标准的 Unix 功能，在大多数系统中都有这两个命令。任务被提交给队列，队列由 *cron* 后台程序处理；任务由与 *crontab* 提交那样的相同限制所控制。*batch* 命令为系统上运行的任务分配优先级。使用 *batch* 命令允许系统管理员根据优先级对任务排序 —— 从高到低 —— 因此允许较重要的任务首先运行。除非系统具有一个结束那些超过指定时间限制的交互任务的机制，*batch* 队列的这一用法依赖于用户是否以合作方式进行工作。在大多数系统下，*batch* 的功能通常被更加复杂的队列系统所代替。你需要从系统管理员那里得到关于 *batch* 和 *at* 队列的可用信息。

at 命令允许在一些指定的时间提交一个任务去运行。只要机器负载低于一个指定水平，以及并行的 *batch* 任务数量仍然没有超过，*batch* 将继续运行这些任务。一旦初始化 *at* 或者 *batch*，直到你使用 Ctrl-D 组合键终止提交，所有的命令行入口都被当作是任务的一部分。就像 *cron* 一样，任何由任务生成的 STDOUT 和 STDERR 都通过电子邮件发送给你，这样你至少可以得到错误状态的提示。下面是命令选项：

-q queuename

指定队列。默认情况下，*at* 使用 “a” 队列；*batch* 使用 “b” 队列。

l

使 *at* 列出当前在指定队列中的任务。

-d jobid

通知 *at* 命令删除一个指定的任务。

-f filename

通知 *at* 从一个文件而不是标准输入来运行任务。

-m

即使在没有生成输出时，也通知 *batch* 和 *at* 发送关于完成的电子邮件。

time

时间可以是 now、teatime、7:00 P.M.、7:00 P.M. tomorrow 等等。查看手册页可以得到更多信息。

下面给出一个例子。假定你想要老板知道你在 3:00 A.M. 正在辛勤工作，只需要在 3:07 A.M. 给他发一封邮件即可。即使你不打算工作到这个时间，也没有问题。在 *at* 命令提示符下，只需要输入下面的命令即可：

```
> at 3:07am
Mail -s "big breakthrough" boss@wherever < /home/jambeck/news
<Ctrl-d>
```

监控磁盘空间的使用和文件的大小

虽然系统上的可用磁盘空间在不断扩展，但是看上去用户总能增加他们的文件来填充它。软件占用了更多的空间；输出文件变得越来越大、越来越复杂；可以创建更多层的分析。由于还没有发明出无限大的数据存储媒体，所以我们还会和磁盘空间的限制问题发生冲突。所以，我们应该能够监控自己使用了多大的空间，并且像在“创建数据的归档”一节中将讨论的那样，如何使数据归档，并把它们保存到合适的媒体上。

使用 du 命令检查磁盘的使用

用法：du -[*options*] *filenames*

du 报告由指定的一个文件或多个文件所使用的磁盘块的数量。在不给出文件名时报告当前工作目录中所有文件所占用的磁盘。 *-s* 标志可以使 *du* 命令仅报告命名文件所占用的空间，而不包括其他文件及其子目录。

使用 df命令检查剩余的磁盘空间

用法： `df`

*df*报告在计算机上的本地和网络文件系统中的剩余磁盘空间。在确定计算机上加载的是什么文件系统时，使用 *df*命令是一种有效的方式。如果到你希望找到的文件系统的连接失败，则那个文件系统就不会出现在 *df*的输出中。*df*的输出格式如下所示：

```
Filesystem             Type    blocks       use     avail %use Mounted on
/dev/root               xfs  17506496  14113592   3392904   81 /
/dev/xlv/xlv_raid_home xfs  62783488  39506328  23277160   63 /scratch-res1
/dev/dsk/dks0d5s0       xfs  17259688  15000528   2259160   87 /mnt/root-6.4
/dev/dsk/dks12d1s7      xfs  17711872  11773568   5938304   67 /ip410
/mnt/local/jmd/balder: NFS server balder not responding
zeus:/hamr              nfs   2205816    703280   1502536   32 /nfs/zeus/hamr
zeus:/hamrscr           nfs   4058200   2153480   1904720   54 /nfs/zeus/hamrscr
zeus:/lcascr1           nfs 142241472 103956480  38284992   74 /nfs/zeus/lcascr1
```

第 1栏是文件系统的实际位置。在本例中，以/开头的为本地目录，那些以一个名称（例如，*/zeus*:...）开头的是另一个机器的物理部分。第 2 栏显示了哪一个协议可以加载远程文件系统 —— 也就是说，哪一个协议可以连接到你的计算机上。下面的三栏分别显示了在文件系统上共有多少单元、有多少单元正在使用、以及还有多少单元可以使用，它们后面的一栏显示了每个设备的使用百分比。最后一栏显示了到文件系统的本地路径。

如果你正在一个由多网络机器组成的系统上工作，知道这些东西十分有用。有时连接会断掉，如前面的例子中的 *balder*。可能在你登录的机器上不能找到你的 home 目录，这是因为 NFS 连接已经断掉。在上述这些情况下，需要指出的是，能够给系统管理员发送一份简洁、有帮助的邮件是很有用的，而不是只会说什么“救我呀！我的 home 目录找不到了。”。

使用 quota检查是否服从系统限额

在一些 Unix系统下，特别是那些为多个用户提供服务的系统下，系统管理员为每个用户都实施了磁盘空间的限额管理。超过磁盘限额的结果可能很明显。你可能会发现你不能写入文件；或者在每次登录到系统时，都会自动提示删除文件。或者，结果可能不提示什么，但是会给你制造麻烦。例如，如果超过了限额，你可能还能够运行一个文本编辑器，但是会发现一个长度为 0 字节的文件覆盖了你正在编辑的文件。或者你的较早的文件开始被删除，为其他用户提供需要的空间。

如果你正在一个共享系统上分享计算机的时间，你会对找出该系统的用户限额、你可以超出该限额多长时间、如果超过限额会出现什么情况以及在哪里和如何归档你的文件很感兴趣。

*quota*命令给出了空间使用和使用限额对系统的限制等基本信息。在大多数 Unix系统下，即使在用户磁盘限额没有超过时，发布 *quota –v* 命令也可以给出磁盘使用信息。

创建数据的归档

这样，经过数月的工作，我们积累了千百万字节的文件和儿层子目录， otter 项目终于完成了。下面该进行另一个全新的项目了。既然是新项目，当然不能只是输入下面的命令了：

```
rm -rf otter/
```

其他人可能需要学习你的发现，或者把它们作为自己研究的起点。从另一个极端来讲，你也不能把文件放得到处都是，或者辛辛苦苦地一次一点把它们复制到另一个位置。并不是在任何时候都需要访问所有的文件；当其他文件更便于保存在其他地方时，有些文件将被替换。本节将介绍 Unix提供的用于归档数据的工具，这样你就不必天天担心它了，但是当你需要这些文件时，还可以找到它们。

tar

用法： `tar` *`functions`* `[`*`options`*`]` `[`*`arguments`*`]` *`filenames`*

在努力合理地设置文件系统之后，看上去丢失存储过程中的结构好像是一种浪费，就像在一次意外的跨国旅行中匆忙地收拾盘子一样。幸运的是，在保留目录结构的时候，有一个 Unix 命令允许对整个文件的目录进行操作。 *tar*把一个目录、它的所有单元文件以及（在你要求的时候）子目录压缩进一个文件，它使用的是被压缩的目录名和 *.tar* 扩展名。*tar* 的选项被分解为两部分：功能（你必须选择一个）和选项。*tar* 是“ tape archive ”的缩写形式，因为这个工具最初的设计目的是用于读取和写入保存在磁带上的归档。 *tar*的另外一种常见的用途是对软件打包，使它可以在因特网上方便地传送。

在运行 *tar* 命令的时候，必须选择下面的一个功能：

c　　创建一个新磁带归档

　　把文件添加到一个现有的归档中

u　　如果文件不存在或者已修改，则把它们添加到归档中

x　　从现有的归档中提取文件

t　　打印归档的目录

tar 命令的可选项如下：

f archive

　　在归档上执行指定的操作，该归档可能是一个设备（例如，一个磁带驱动或者一个可移动磁盘）或者一个 *tar* 文件

v　　（verbose模式）打印每个已归档或提取的文件名，在操作时使用一个字符表示功能（*a* 表示归档，*x* 表示提取）

w　　（whiny 模式）在每一步都会要求确认

注意，功能和选项都不需要通常出现在 Unix 命令选项前面的连字符。

如果输入下面的命令：

```
% tar cvf otter/
```

则 *otter/* 目录和所有子目录都会被卷进一个名为 *otter.tar* 的文件。使用 *v* 选项是一个很好的实践，这样你就可以看到，当执行归档的时候，是否会发生严重的错误。

另一方面，如果想在磁带驱动 *nftape* 上创建 *otter/* 目录的归档，可以输入下面的命令：

```
% tar cvf /dev/nftape otter/
```

依次会出现一对关于 *tar* 的警告消息。首先，在自己的系统上使用 *tar* 之前，应该使用 *which* 明确 GNU 或者标准的 *tar* 版本是否已经安装。几种选项在每个版本中的意义不同；前面列出的几个选项在每个版本中都是相同的。

第二，你所创建的 *tar* 文件将和它下面的目录及子目录的所有内容一样大。如果归档后的目录很大，然而磁盘空间有限，或者需要移动大量的归档数据，就会产生可怕的后果。在这种情况下，应该把目录分割为更方便管理的子目录，然后再压缩这些子目录。

如果在当前的文件系统或分区上没有空间同时保存文件和为该文件创建的归档；或者想下载一个完整的归档文件，然后解开它以找回一些文件，可以结合使用 *ftp* 和 *tar* 命令在网络上或者仅仅在不同分区之间传送归档。如果所涉及的文件数量很大，使用这种方法发送一个归档到目标中会比使用 *cp -r* 命令花费的时间少。*ftp* 程序可识别出用命令替换输入文件名的形式。该命令在本地机器上的子 shell 中执行，并且在本地文件系统上操作文件。其语法如下：

```
ftp command "| command" filename
```

在 *ftp* 程序中，下面给出了如何发送 *tar* 命令的输出，用引号把 *tar* 命令包围起来，指定的 filename 作为远程机器上的目标：

```
put "|tar cvBf - *" filename
```

下面是如何通过 *tar* 命令引导下载的归档，其结果是，只有在归档中的指定目录下的文件被提取：

```
get filename.tar "|tar xvf - dirname"
```

最后，下面是如何列出远程归档的内容：

```
get filename.tar "|tar t - *"
```

compress

用法：compress -[*options*] *filenames*

最后，除非更加易于管理，否则我们并不想留下大型的压缩文件来弄乱自己的文件系统。在这种情况下，数据压缩工具就显得很重要了，因为它们可以使文件占用较少的硬盘空间。*compress* 是一个标准的 Unix 文件压缩命令。它和第三章中所使用的 *uncompress* 命令是相对的，*uncompress* 命令的作用是解开文档和软件的压缩。*compress* 命令在文件名的最后添加一个 *.Z* 扩展名。

下面是 *compress* 命令的一些最有用的选项：

-f 强制压缩；即使已经有了文件的一个压缩版本，主要是不会覆盖现有的压缩文件

-v （verbose 模式）打印出文件所达到的压缩百分比

-r （recursive 模式）如果对一个包含子目录的目录进行压缩，则压缩子目录的内容及其原始目录的内容

如果你有一个名为 *stoat.txt* 的文本文件和来自最后一个部分的 *otter/* 目录下的 *tar* 文件，并且你想把它们压缩后查看所达到的压缩比，可以输入下面的命令：

```
% ompress -v stoat.txt otter.tar
```

该命令生成两个文件：*stoat.txt.Z* 和 *otter.tar.Z*。可以使用 *uncompress* 命令或者 *gzip -d* （后面会介绍）解压缩该文件。令人惊奇的是，对自然语言（人类使用的语言）的压缩比大约是 60%，而对编程语言的压缩比大约是 40%。你可以试着压缩一下自己所喜欢的蛋白质序列，看一下所能达到的压缩比：压缩比的值可能会非常不同，这依赖于在序列中是否有重复。

gzip

用法：`gzip -[options] filenames`

通常情况下，作为对标准 Unix 压缩的补充，还有一种更快、更有效的 GNU 工具：*gzip*。*gzip* 和 *compress* 的工作方式非常相似，但是比较起来，它的压缩比更高，因为它使用的是一种超级算法。*gzip* 为它所压缩的文件添加一个 *.gz* 扩展名。它包括前面所提到的 *compress* 的选项，并且增加了下面的一些新选项：

-N （默认设置）从被压缩的文件保存原始的名称和时间戳

-q （quiet 模式）在运行的时候，不发出警告

-d 返回一个被 *gzip* 压缩的文件到它的未压缩状态； *gzip* 也可以识别和解开由 *compress* 产生的文件。

第三部分

生物信息学工具

第六章，在 Web 上进行生物学研究

第七章，序列分析、成对比对以及数据库搜索

第八章，多序列比对、进化树和简图

第九章，蛋白质结构的可视化和结构性质的计算

第十章，根据序列预测蛋白质的结构和功能

第十一章，基因组学和蛋白质组学工具

第六章

在Web 上进行生物学研究

因特网彻底改变了科学家查找和交换信息的方式。原来必须以纸张为媒介交流的数据现在已经数字化，并从中心数据库发布。杂志也可在线发表。基本上每一个课题组都具有一个网页，提供包括复制、下载软件、查找数据以及数据的自动处理等各项服务。

对于单词‘ bioinformatics ’进行一个简单的Web查找就会产生成千上万条信息。你想要的信息可能是第345项，或者根本没有找到。从哪里才能够只得到有用的软件、数据以及科技文章，而没有无用的信息呢？总是通过一个简单的Web 查找不会得到这种理想的结果。

怎样判断到底哪一条信息是有用的呢？在Web上的出版物对于每条信息都给出了权威性的评价。怎样判断软件是否给出了你想要的结果，以及正确完成其功能呢？

在本章中我们将研究在Web上查找信息的艺术。包括搜索引擎以及查找过程，从何处可找到科技文章及软件，以及怎样应用经典的在线信息资源，如 PebMed。找到所要信息的位置后，通过本章的学习可以帮助你怎样利用这些信息。对于生物学家来说，最大的信息资源是公共的生物学数据库。我们将讨论公共数据库的历史、数据注解、数据可采用的各种形式，以及怎样输入及输出数据。最后，我们会给出一些建议，帮助你确定查找出的信息的质量好坏。

对于生物学研究来说，因特网是一个非常有用的信息资源。除了允许研究工作者非

常容易地交流软件及数据外，还是为计算机软件、硬件、实验方法、步骤以及购买实验室仪器提供实用性建议的一种资源。我们应该好好地利用因特网，而且聪明地利用它来服务自己的工作。

应用搜索引擎

AltaVista、Lycos、Google、HotBot、Northern Light、Dogpile 以及许多其他的搜索引擎可以帮助我们从组成 Web 的上百万个网页中查找出符合需要的东西。然而，作为一个科学家，我们正在寻找的不是一般的 Web 物品，如，在Web 上订购书籍，或在线新闻。我们正在寻找的东西就像大海捞针一样困难。

作为一个科学家，知道怎样建立一个查询，并在查询时清除大多数可能遇到的废物是非常有用的，因为这种情况无论是在 Web查找还是在以关键词为基础的数据库搜索时都会遇到。理解怎样设计可以限制搜索的布尔查询是一个非常关键的研究技巧。

布尔搜索

许多 Web 冲浪者的查询都是无计划的。在小方框中键入一些关键词，并观察产生什么样的结果。但是每一个搜索引擎都有不同的默认假设，因此，如果在 Excite 的查询栏中键入“ Protein Structure”， 得到的查询结果和在 Google 查询栏中键入“Protein Structure ”得到的查询结果相比，可能会完全不同。为了进行有效的搜索，需要应用布尔逻辑。这是一种非常简单的方式，用来表达一组事物怎样划分，又怎样结合。

搜索引擎都应用一些布尔逻辑，像大多数公共生物学数据库的查询形式一样。布尔查询通过用运算符 AND、OR及 NOT连接的一系列查询词组来限制从数据库得到的结果。

这些运算符的意思非常明显：当把两个关键词用 AND连接时，查找到的文件应该包含关键词*1* 和关键词 2; 当把两个关键词用 OR 连接时，查找到的文件或者包含关键*词1* 或者包含关键*词2*（或者都包括）当把两个关键词用 NOT 连接时，查找到的文件将包含关键词 *1* 但不包括关键词 2。

然而，不同的搜索引擎在解释空格或隐含的操作符时有所不同。一些搜索引擎认为空格表示 OR，因此键入“ protein structure ”时，实际上寻找的是“ protein ”或“structure”。如果在 Excite 上查找“ protein structure”，因为 Excite 默认的是 OR，我们会得到有关时尚的食物及蛋白质的许多广告以及有关的补充，但这些结果与我们感兴趣的科学站点无关。另一方面，用 Google 搜索默认的是 AND，因此只能查找到同时包括蛋白质和结构的文献，这些可能是我们初步要的文献。在设计查询前，首先应该了解使用的搜索引擎是怎样工作的，这一点是非常必要的。

布尔查询是按照从左到右的顺序进行的，像阅读文章一样。使用括号可以建立更复杂的布尔查询。比如，如果要查询的文档包含关键词 *1* 以及关键词 *2* 或关键词 *3* 中的任意一个，但是不包含关键词 *4*，则查询语句应该是这样的：（关键词 *1* AND（关键词 *2* OR 关键词 *3*））NOT 关键词 *4*。

许多搜索引擎允许使用引号来指定一个短语。例如，如果在想要查询的文档中，“protein structure ”这个单词是在一起顺序出现的，则搜索“ protein structure ”会适当地缩小搜索范围。

在一个文献数据库中搜索有关计算蛋白质分子的静电势能的文献时，可能只想查找作者 Barry Honing 和 Andrew McCammon 的文献。这时，可以建立下面的一个布尔查询结构：

```
((protein AND "electrostatic potential") AND ( Honig OR McCammon))
```

这个语句会通知搜索引擎去查询那些包含单词“ protein ”及短语“ electrostatic potential”，以及人名为 Honig 或 McCammon 两者之一的文献。

有许多关于布尔搜索的优秀网站。在 *www.goole.com* 中，使用 Google 搜索引擎搜索短语 *boolean searching*，看看会得到什么搜索结果。

搜索引擎算法

本书的目的不是详尽无遗地描述搜索引擎的工作原理。搜索引擎在建立数据库和分级站点方面有明显的不同。这些不同使得一些搜索引擎在搜索科学和技术 Web站点方面比其他的搜索引擎要有用得多。

Web搜索引擎数据库建立和检索策略的关键功能是可以提供免费的URL提交、全文检索 、自动广泛的网上搜索 、快速的更新率以及对结果的合理分级策略。

当前我们最喜欢用的搜索引擎是Google，Google包括的范围是极其广泛的，可以在十亿多 URL 上进行检索。网页的分级是根据本页内容被其他页链接的次数确定的。与有许多网页链接的链接比孤立的网页的链接更重要。 Google搜索将把我们引到与要搜索的主题相符的最易浏览的网页，它将这个工作做得很好。 Google会缓存 Web页的副本，所以即使服务器脱机，也能访问这些网页。当再次访问时，只是返回包含所有相关查询项目的页面。 Google使用标准布尔搜索格式的简单版本，允许特定的服务，如定位所有与感兴趣的网页链接的网页。

然而对一般的用户来说， HotBot可能是最好的搜索引擎。HotBot搜索的范围较广，可以定期更新，而且可提供以表单为基础的查询工具，减少了简单查询语句格式的需要。

查找科学文献

传统上，科学家相信在印刷期刊上发表的论文的质量，因为这些期刊是可以参考的。编辑把每篇文章发送给专家评委组 ,这些评委都有资格评判研究者所写的论文质量，在论文允许发表之前，这些评委会在其手稿上发表自己的意见。这些论文常常需要补充、修改，有时甚至需要更深入的实验进行补充。科学方面期刊的出版同时也常常伴随着电子期刊的出版。几乎每种主要的期刊都有自己的 Web站点，虽然其文摘可免费得到，大多数期刊的电子版只给发送预订者。网上期刊的科学论文如同印刷出版物一样要经历严格的评审程序。

另一趋势是没有印刷版本的电子期刊，这些期刊可以经常参考，并且发现它们的出处并不难。例如，《Journal of Molecular Modeling》 是由 SpringerVerlag 出版的电子期刊，在它的首页上链接着有关杂志编辑规定的信息。

在生物科学领域查找科学文献的一个极好的资源，是 NCBI发起的国家医药库的免费服务器。这个服务器使任何人用 Web 浏览器搜索 Medline数据库成为可能。也有其他质量较好的文献数据库，但大多数都是收费的。一般用户所在的研究单位能够访问 Lexis-Nexis 或 Cambridge Scientific Abstracts这些资源。

然而除了上面提到的资源之外，任何入都可以在 Web上发布信息，研究小组常常在他们的网站上发布作为技术报告的论文。这些技术报告可能永远不会被仔细评论，也不会在研究团体之外的地方发表，判断其质量可靠性的惟一线索，是作者的声誉和专业。这并不是说不能相信或者不能查找这些资源，在许多政府组织和学术研究团体的网站上有许多接近教科书品质的参考材料。比如，华盛顿大学基因组中心在基因组测序方面就有一个极好的自学材料， NCBI在应用 BLAST序列比对程序及其变体方面有很好的实用指导材料。

有效地使用 PubMed

在生物学家可利用的科学文献中， PubMed（*http://www.ncbi.nlm.nih.gov/entrez/query.fcgi*）是最有价值的 Web资源之一。在 PubMed上可索引 4000 多种杂志，包括大多数的细胞分子生物学、生物化学、遗传学和相关领域的较受推崇的杂志，以及医学专业人员感兴趣的一些临床杂志。

PubMed使用以关键词为基础的搜索策略，在查询语句中允许有逻辑运算符 AND、OR和 NOT。用户可以根据在方框的搜索项中附上一个查找范围，来列出对每一个搜索项进行查找的数据库范围。

另外，用户可以使用 MeSH（Medical Subject Heading）查找 PubMed。MeSH 是一个标准词库，可用来帮助查找用不同名称代表同一概念的文献。 MeSH 浏览器（*http://www.nlm.nih.gov/mesh/meshhome.html* ）允许用户在键入一个单词或单词片段后在 MeSH库中找到相关的关键词。PubMed可以自动寻找与查询项相关的MeSH词，并使用它们提高查询的质量。

例如，在 PubMed 中寻找‘ protein electrostatics”。除非特别指出，“protein ”和“electrostatics ”是用 AND 自动结合在一起的。提交给 PubMed 的布尔搜索语句实际上如下所示：

```
((("proteins"[MeSH Terms] OR protein[Text Word]) AND ("electrostatics"[MeSH Terms]
 OR electrostatics[Text Word])) AND notpubref[sb])
```

搜索结果如图 6-1 所示。

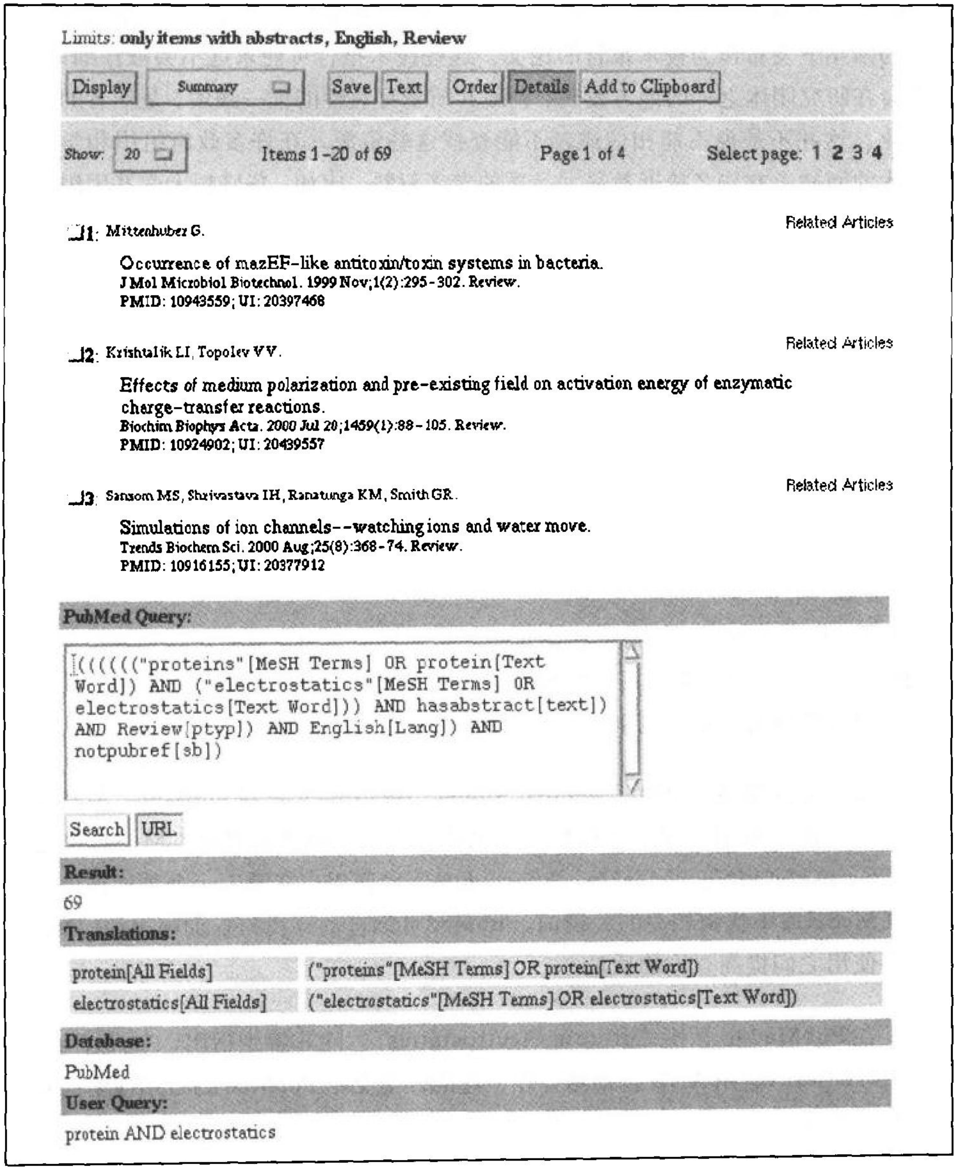


图6-1　PubMed 搜索的结果

由图 6-2 可以看到，PubMed 还可以使用 Web 界面缩小搜索范围。在 PubMed 网页查询框下面的 Limits 链接可将我们带至这个 Web 表单。

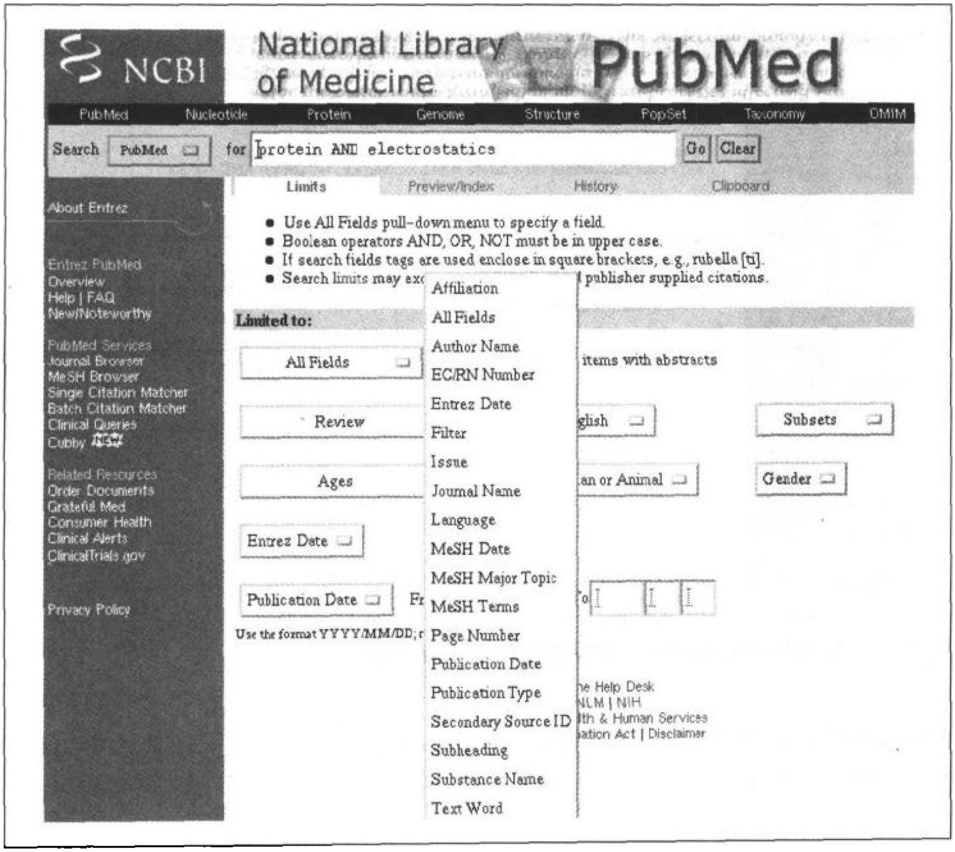


图 6-2　在 PubMed 中使用 Limits 菜单缩小搜索范围的策略

Limits 表单允许在查询上加上特殊条件。可以把搜索范围限制在 PubMed 数据库记录中特定的字段，如，Author Name 或 Substance Name 字段。也可以通过语言、内容（比如，只搜索评论文章或者临床试验文章）和日期来限制搜索。对于临床研究的文献，搜索时可以根据种系、年龄和研究对象的性别进行限制。

Preview/Index 菜单允许建立一个详细的交互式查询。可以选择一个特定的数据字段（比如，Author Name 字段），然后输入只在这个特定字段内要搜索的词，点击 AND、OR 或者 NOT 按钮，把新词加到原来的搜索词上。

例如，我们可能对 “protein AND electrostatics” 开始进行一个普通的搜索，然后进入 Preview/Index 页（见图 6-3）并只在 Author Name 字段中搜索 “Glison OR McCammon”。

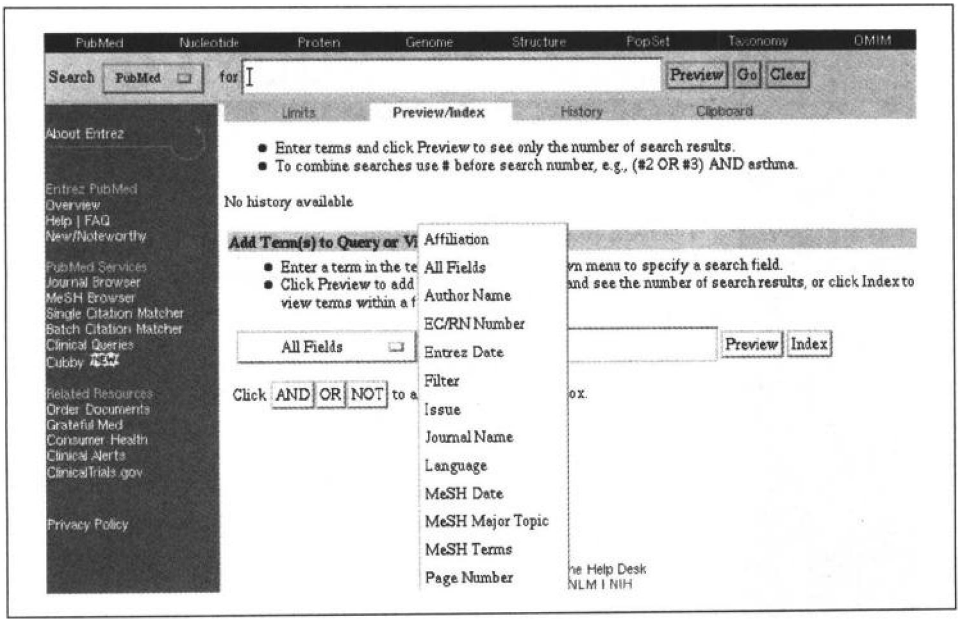


图 6-3　　使用 Preview/Index　表单建立 PubMed 查询

也可以在History 表单中使用选项以访问以前搜索的结果，并且可以在查询中加入新的词语以缩小搜索的范围。

如果想收集多个查询的搜索结果，并在一个大文件中保存这些结果，那么可使用剪贴板。为了将单个结果保存到剪贴板上，直接点击要保存的结果附近的复选框，然后在结果页面上面的菜单中点击添加到剪贴板按钮（注 1）。

如果找到了一个有助于在 PubMed 上运行的搜索策略，那么，可以以 URL 的形式保存这个策略，然后通过 URL 语句在将来的任何时候重复进行同样的搜索。为了保存 PubMed URL ，可在结果页面上点击 Details 链接，然后在 Details 页面上点击 URL 链接。搜索的 URL 就会显示在 Web 浏览器上部的 Location 字段中，可将其作为书签。

注 1：　可以看到，如果正在处理许多结果页面的话，点击所有用来选择和保存单个结果的复选框会很费时。如果采用一种搜索策略肯定只生成一个我们需要的结果的话，就容易多了。在 NCBI 工具中没有相应的解决方案，而自己编写脚本处理批量的结果又未必可行。计算机程序的局限性正在于不能很好地分析人类语言。

对 PubMed 搜索来说，可用作书签的 URL 如下所示：

```
http://www.ncbi.nlm.nih.gov:80/entrez/query.fcgi?cmd=PureSearch&db=
PubMed&details_term=%28%28%28%28%28%28%22proteins%22%5BMeSH%20Terms
%5D%20OR%20protein%5BText%20Word%5D%29%20AND%20%28%22electrostatics
%22%5BMeSH%20Terms%5D%20OR%20electrostatics%5BText%20Word%5D%29%29
%20AND%20hasabstract%5Btext%5D%29%20AND%20Review%5Bptyp%5D%29%20AND
%20English%5BLang%5D%29%20AND%20notpubref%5Bsb%5D%29
```

花几小时的时间，开发一些可为你所用的详细的 PubMed 搜索策略，并保存它们，可为将来节省许多时间。

公共的生物学数据库

在生物学的分子级的命名法问题是一个广泛存在的问题。基因通常以非系统命名的名字而为大家所熟知。这些可能来源于模型系统中的发育生物学研究，因此一些基因由于在特定的动物发育中的作用，命名为*flightless*、*shaker* 和 *antennapedia* 。其他一些名字由细胞生物学家选择，代表基因在细胞级的功能，如 *homeobox*。还有其他一些名字由生物化学家和结构生物学家选择，代表在基因发现之前蛋白质已经分离及研究的情况。尽管蛋白质是基因的直接产物，它们并不经常与编码基因具有相同的名字或代码。这种令人迷惑的命名法通常意味着只有研究一个特定的基因、基因产物或者生物化学过程部分的科学家可以立即认出一般的基因名称指的是什么。

一个简单的有机体的生物化学比 Linnaeus 时代已有物种的分类法所包含的信息更加复杂，因此不能指望能轻易得到清楚而广泛的命名系统。对于一个给定的基因，有许多信息需要了解：它来源的机体、染色体的位置、激活子序列的位置以及调节开关的调节蛋白质的鉴定；基因可以根据在生物发育中开启的时间以及表达的组织来分类；也可以由它们的产物的功能来分类，无论它们是结构蛋白质、酶或者功能 RNA；也可由其代谢产物参与的代谢途径来分类，其产物是代谢途径的部分环节，也可以由它修改的底物或者是它的产物来分类；还可以由基因蛋白质产物的结构来分类。显然，这里有许多信息可以收集浓缩为一个合理的命名法。图 6-4 显示了一部分与一个基因相关联的信息。

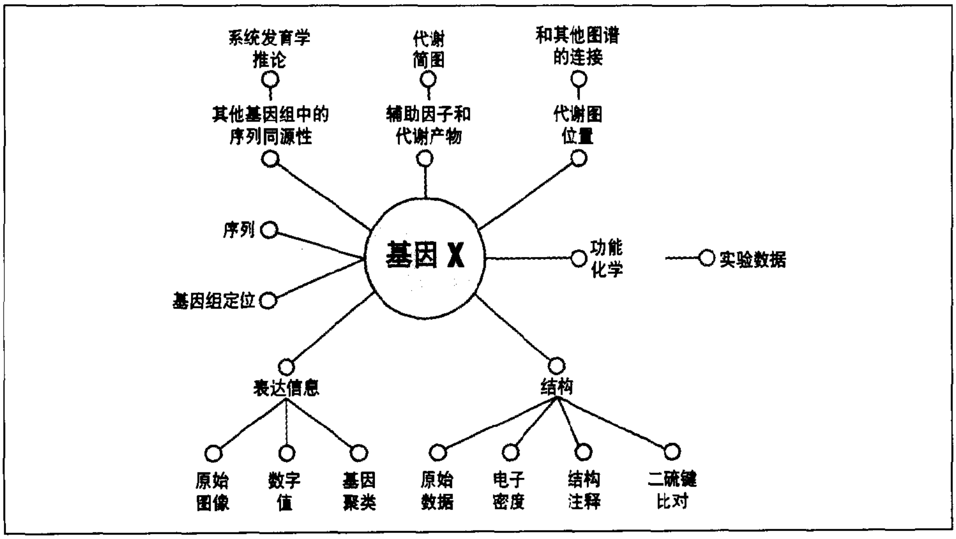


图 6-4　与单个基因相关的一些信息

生物数据库维护人员的问题主要是注解，即在数据库中加入充足的信息，使得在此数据库中，即使确实有一个含义模糊的普通的名字，它是什么基因也毫无疑问，并在信息、基因序列和序列号之间建立正确的链接。基因数据的正确注解本身就是一个活跃的研究领域，因为研究者试图发现通过基因组传递信息的正确方式。

关于命名法的问题，存储在电子数据库中的大分子数据给了一个正确的解决方法。其解决方法是为每一个进入数据库的成员编码一个序号，然后存储在相关的关系数据库中，其序号代表基因的名字或基因产物及所有有关基因的信息。这种策略是现代生物数据库中常用的方法。数据库必须解决的问题和开发命名法存在的问题一样。然而，通过使用关系数据库和复杂的查询策略，他们（可能是有些不幸地）忽略了重要问题：为科学家寻找一个简便的非数字识别基因的方法。

数据注释和数据格式

在生物信息学中，生物学数据的表示和发布也是一个正在研究中的问题。 DNA和RNA的核苷酸序列和蛋白质的氨基酸序列简化为一个字符串，其中仅用一个字母代表一个核苷酸或氨基酸。在代表序列的数据库中剩余的问题是证实数据的正确性，对数据进行全面的注释，对越来越大的数据库进行处理，如染色体和整个基因库的序列这样的数据库。

标准的简化的生物分子的三维结构表示法由分子中原子的笛卡儿坐标组成。分子的这种表示十分直接；另一方面，在结构数据库中，还有许多复杂的问题没有完全解决。尽管生物学术界正在达成一种共识，以确定什么样的结构注释是需要的，但是注释仍然是结构数据的一个问题。

在过去15年中，不同的研究人员在报道生物学数据方面已形成自己的风格及格式。生物学序列和结构数据库在美国和欧洲得到并行的发展。使用专有软件进行数据分析有助于许多专有的数据形式的混合。虽然也有许多专门的数据库，在此，我们将着眼于维护整个数据类的综合的数据库。

三维分子结构数据

尽管DNA序列、蛋白质序列和蛋白质结构在某种意义上只是以不同的方式来代表相同的基因产物，但是这些数据类型常常以独立的数据库及不相连的数据格式来保存。这主要是由于序列和结构确定方法在形成过程中各有自己的历史。

在建立公共的 DNA序列数据库之前的10年中，建立的第一个公共的分子生物学数据库是 PDB （蛋白质数据银行），是蛋白质分子的 X 射线晶体结构的中心知识库。

虽然第一个完整的蛋白质结构是在 20世纪 50年代发现的，但直到 70年代末期可获得的蛋白质结构仍不太多。计算机还未发展到使蛋白质结构坐标数据的图表表达成为可能的程度，但至少是以有用的速度在进行。然而，1971 年，在 Brookhaven National Laboratory建立了 PDB ，在一种以计算机为基础的档案中存储蛋白质结构数据。开发数据格式很大程度上是为了适应早期计算机技术的需要。在 20世纪 70年代到80 年代，PDB在不停地发展，从1973 年的 15个坐标集合发展到 1976年的 69个，直到 1988年，每年保存的坐标集合的数目保持在 100以下，直至 1988年，PDB 条目都少于400 个。

1988~1992年，PDB 的发展在它的指数生长曲线中处于转折点。到1994 年1月，在PDB中有2143个条目；到写此书时为止，PDB已经基本达到 14000个条目。对 PDB的管理已转移至大学及公共机构的研究者合作机构，称为Research Collaboratory for Structural Bioinformatics （结构化生物信息学研究合作者），记录结晶学数据的一种新的形式，mmCIF（Macromolecular Crystallographic Information File，大分子晶体信息文件）正在替代陈旧的 PDB形式。现在杂志上要发表的结晶学结果，需要

提交给PDB 作为出版条件，这意味着由科研工作者得到的几乎所有蛋白质结构数据都可以及时在 PDB 上找到。

蛋白质结构的数据驱动研究的结果一般是多余的，并且缺少 PDB 的广泛性。有许多蛋白质结晶结构已经输入至数据库。在任何蛋白质结构的统计学研究中，选择 PDB 数据的子集，是非常重要的一步。到 1998 年 12 月，在 PDB 中只有大约 2800 个蛋白质链与其他的完全不同（序列常见的部分少于 95%），被认为是独特的结构。许多蛋白质结构的统计学研究，是根据只占蛋白质序列 25%的蛋白质链部分进行的，如果应用这个标准，将只有 1000个独特的蛋白质折叠表示在 PDB 中。随着可利用的生物学序列数据数量的增长， PDB 的发展将远远落后于基因序列数据库的发展。

DNA、RNA 和蛋白质序列数据

一般来说，序列数据库专门收录一种类型的序列数据： DNA、RNA或蛋白质。在欧洲、日本和美国有主要的序列数据收集和存储的站点，有独立的团体为在主要的公共数据库中收集到的所有数据提供镜像，并经常提供一些软件来使这些数据增值。

1970年，Ray Wu 为第一个 DNA片段测序，在环形的 DNA 中使用酶使之开环后，在 DNA 末端有 12个碱基以一串的形式出现。然而，DNA测序的证实比蛋白质测序要困难得多，因为没有一个化学过程可以选择性地切开核苷酸链上的第一个核苷酸。当 Robert Holley 报道从酵母中得到一个76个核酸 RNA 分子的序列时，已经是在七年之后。在 Holley 的序列发表后，其他的研究团体改进了测序的规程，甚至可成功地对一个 3200 碱基的噬菌体基因组测序。 DNA 测序真正的进步是在 1975 年之后，使用的是 Allan Maxam 和 Walter Gilbert 设计的化学裂解方法，以及 Frederick Sanger 的链终止子方法。

在 1979年建立的第一个DNA序列数据库是Los Alamos 国家实验室的GSDB（Gene Sequence Database，基因序列数据库），由于世界范围的合作，现在 GenBank 代替了 GSDB。迄今为止，基因序列信息仍旧可以通过 National Center for Genome Resources （美国国家基因组资源中心）从 GSDB 中获得。

欧洲的 Molecular Biology Laboratory （分子生物学实验室），日本的 DNA Database（DNA数据库）以及 NIH（National Institutes of Health ）合作使得通过 GenBank 可获得所有可获得的公共的序列数据。 NCBI 已经为序列数据开发了一个标准的关

系数据库格式,即 ASN.1 格式。这种格式不仅使得在 GenBank 中定位正确的种类的正确序列变得更容易，而且可为访问非冗余的数据库版本提供许多服务。

在DNA 序列数据库发展的第一个十年中，其发展速度缓慢。在 1992 年，GenBank 只包含 78000 个 DNA序列 —— 只有 1亿多个 DNA碱基对。在 1995 年，由于人类基因组计划的启动和 DNA 测序技术的发展，GenBank 中的数据随着快速增长。现在每隔 6~8 月，GenBank 的容量都要翻一番，它的增长速率正在平稳增加。

基因组数据

除了人类基因组计划外，还有许多独立的模式生物的基因组计划数据库。 GenBank 可代表基因组计划数据库的序列内容，但是基因组计划站点也为从事该生物研究的工作者提供从基因组图谱到相关资源的任何资料。到 2000年 10月份 ,NCBI的 Entrez 基因组数据库包含超过 900 种部分或完整的基因组。其中有许多是病毒。其他部分包括细菌、古生物、酵母；一般研究的植物模型系统，如，拟南芥、水稻以及玉米；动物模型系统包括 C.elegans、果蝇、小鼠 、大鼠以及河豚；还有细胞器的基因组。NCBI 访问这些数据库的基于 Web 的软件工具正在不断发展，并且变得越来越复杂。

生物化学途径数据

最重要的生物学活动不是由单个的分子来完成的，而是多个分子和谐完成的。在 20 世纪早期，生物化学家已研究酶及其底物的整体功能。一些研究团体开始在数据库中组织存储这些途径。途径数据库的两个例子是 WIT 以及 KEGG。WIT 是“ What Is There? ”的缩写，是 Argonne 国家实验室开发的。它是包含基因组已完全测序的有机体重新建立代谢途径的数据库。 KEGG（Kyoto Encyclopedia of Genes and Genomes）包含相似的数据，但是和序列、结构及遗传数据库相链接。两个数据库都可通过 Web 界面查询，可通过自动化和人类专门技能的结合来管理。

除了这些整个基因组的“ 部分目录 ”之外，其他专业着眼于特殊途径的数据库也已经在开发（如细胞间信号或通过微生物降解化学复合物）。

基因表达数据

DNA微阵列（或基因芯片， gene chip）是研究基因表达的微型实验室。每个芯片包含精确设计的探针分子的阵列，这些阵列中的探针分子可与特定种类的DNA或mRNA 片绑定。用荧光分子标记DNA或RNA，可允许在细胞中的任何基因的表达水平被量化测定。在分子生物学中，微阵列还有其他方面的应用，但是在研究基因表达的应用方面，已经开辟了测定基因组功能的新途径。

随着90年代末期DNA微阵列技术的发展，可获得的基因表达数据的增加最终将与序列和结构数据库并行增长，这一点变得越来越明显。因此，微阵列数据是可获得的公共原始数据的另一种数据类型。在可选择性的数据库中，原始的微阵列数据开始成为公共可获得的数据，对于建立一个这样数据的中心的数据仓库的讨论正在进行中，然而，可传送这些数据的格式还未标准化；通常，这些数据以大的电子数据表或以制表符界定的文本形式出现。微阵列数据两个最大的资源是National Human Genome Research Initiative的Microarray Project站点以及斯坦福基因组资源站点。因为许多早期的微阵列实验是在斯坦福大学进行的，因此，他们的基因组资源站点既可以链接到原始资料，在有些情况下也可链接到可使用基因的名字或功能描述进行查询的数据库。最近，欧洲生物信息学研究所（European Bioinformatics Institute）已在开发一套在数据库中存放微阵列数据的标准。一些数据库也存有二维凝胶电泳结果，包括 SWISS-2DPAGE 和 HSC-2DPAGE。2D-PAGE 是一项可以同时对细胞中许多蛋白质的浓度进行定量研究的技术。

这两项技术的结合是理解基因组如何发挥作用的强有力的工具。表 6-1 总结了我们在本节讨论的一些重要的数据库的 Web 资源。

表 6-1　主要的生物学数据和信息资源

主题	资源	链接
生物医学文献	PubMed	*http://www.ncbi.nlm.nih.gov/entrez/query.fcgi*
核酸序列	GenBank	*http://www.ncbi.nlm.nih.gov:80/entrez/query.fcgi?db=Nucleotide*
	EMBL/EBI 的 SRS	*http://srs.ebi.ac.uk*
基因组序列	Entrez Genome	*http://www.ncbi.nlm.nih.gov:80/entrez/query.fcgi?db=Genome*
	TIGR 数据库	*http://www.tigr.org/tdb/*

表6-1　主要的生物学数据和信息资源(续)

主题	资源	链接
蛋白质序列	GenBank	*http://www.ncbi.nlm.nih.gov:80/entrez/query.fcgi?db=Protein*
	ExPASy的SWISS-PPOT	*http://www.expasy.ch/spro/*
	PIR	*http://www.nbrf.georgetown.edu*
蛋白质结构	Protein Data Bank	*http://www.rcsb.org/pdb/*
Entrez结构DB		
蛋白质和肽质谱	PROWL	*http://prowl.rockfeller.edu*
翻译后修饰	RESID	*http://www.nbrf.georgetown.edu/pirwww/search/textresid.html*
生化与生物物理信息	ENZYME	*http://www.expasy.ch/enzyme/*
	BIND	*http://www.ncbi.nlm.nib.gov:80/entrez/query:fcgi?db=Structure*
生物化学途径	PathDB	*http://www.ncgr.org/software/pathdb/*
	KEGG	*http://www.genome.ad.jp/kegg/*
	WIT	*http://www.wit.mcs.anl.gov/WIT2/*
微阵列	Gene Expression Links	*http://industry.ebi.ac.uk/~alan/MicroArray/*
2D-PAGE	SWISS-2DPAGE	*http://www.expasy.ch/ch2d/ch2d-top.html*
Web资源	The EBI Biocatalog	*http://www.ebi.ac.uk/biocat/*
	IUBio Archive	*http://www.iubio.bio.indiana.edu*

搜索生物学数据库

在Web上有许多生物学数据库，有许多可选择的Web界面可提供进入相同的数据集的路径。可根据自己的需要选择使用哪一个，但是需要明白各种数据类型的中心数据仓库，以及外围数据库相对于中心数据库的更新频率。

尽管保存新的生物学数据类型的数据仓库有很多，在这里我们集中说明的是两个已建立的数据库：NCBI的 GenBank，是 DNA 序列的数据库；Protein Data Bank，是分子结构数据库。每一个数据库都有自己的存储步骤，对于新的数据类型，存储的步骤还未建立完备，变化得比较快。然而， NCBI 及 RCSB 是一个成熟的、自动化的、以 Web 为基础的存储系统，在近期不可能有很大幅度的改变。

GenBank

NCBI与 EMBL和其他国际组织合作，提供的 DNA 序列数据是世界上最完整的数据库，也提供 PubMed 这个分类数据库以及蛋白质序列和结构数据的其他访问点。这个叫做 GenBank的数据库，可以从这个网址访问：*http: //www.ncbi.nlm.nih.gov:80/entrez/query.fcgi?db=Protein*。

NCBI 包含各种有机体、各种来源、各种类型 DNA 的序列数据，从 mRNA到 cDNA 克隆、表达的序列标签（EST）、高通量的基因组测序数据和序列多态性的信息。NCBI 数据库的用户需要明白这些数据类型之间的差异，以搜索到最适合其工作的数据类型。在所有 GanBank 搜索中，可能遇到的主要序列的类型包括：

mRNA

信使RNA，是基因组 DNA 的转录产物。mRNA 可以通过去除内含子(intron，在真核细胞中）或其他方式被细胞剪辑，其结果导致与转录的基因组 DNA 不同。或许是“部分”或“全部”， mRNA 可能并不覆盖基因的全部编码序列。

cDNA

DNA 是通过 mRNA 的反转录，人工产生的 DNA序列。cDNA粗略地代表产生 mRNA 的基因组 DNA 的编码部分。也可以是“部分”或“全部”。

基因组 *DNA*

从基因组测序获得的 DNA序列包含编码和非编码的 DNA序列。也许包含内含子，重复序列和其他部分。基因组 DNA （与基因组检测序列相对）一般是“全部”；它是对一段基因组链进行多次测序的结果，一般可以认为比较好地代表了此区域的真正 DNA 序列。

EST

在特定的条件下或特殊的发育阶段，从细胞中提取的mRNA制备的短 cDNA序

列（如 arabidopsis thaliana 2周苗或巴伦西亚桔种）。EST 也用于快速的基因鉴定，不会覆盖基因的全部编码序列。

GSS

GSS是指基因组检查序列。直接来源于基因组计划的单次测序。每个区域的序列只覆盖一次，可能包含相对较多的测序错误。如果正在寻找的是基因组计划中正在研究的新假说的基因注解，应该在搜索中包括 GSS。

在搜索 GenBank时有两种方式。第一种是使用基于文本文件的查询来搜索数据库中与每一个 DNA序列条目相关的注解。第二种我们将在第七章讨论，是在序列数据库中使用一种称为 BLAST的方法比较查询 DNA（或蛋白质序列）。

这里有一个 GenBank 记录的例子。每个 GenBank 条目除序外之外还包含注解：基因鉴定的信息、描述该特性时的条件等等。

```
LOCUS       AB009351 1412 bp    mRNA    PLN        22-JUN-1999
DEFINITION  Citrus sinensis mRNA for chalcone synthase, complete cds, clone
            CitCHS2.
ACCESSION   AB009351 VERSION AB009351.1 GI:5106368
KEYWORDS    chalcone synthase.
SOURCE      Citrus sinensis young seed cDNA to mRNA, clone:CitCHS2.
  ORGANISM  Citrus sinensis
            Eukaryota; Viridiplantae; Streptophyta; Embryophyta; Tracheophyta;
            euphyllophytes; Spermatophyta; Magnoliophyta; eudicotyledons; core
            eudicots; Rosidae; eurosids II; Sapindales; Rutaceae; Citrus.
REFERENCE   1 (sites)
  AUTHORS   Moriguchi,T., Kita,M., Tomono,Y., EndoInagaki,T. and Omura,M.
  TITLE     One type of chalcone synthase gene expressed during embryogenesis
            regulates the flavonoid accumulation in citrus cell cultures
  JOURNAL   Plant Cell Physiol. 40 (6), 651-655 (1999)
  MEDLINE   99412624
  [...]
FEATURES    Location/Qualifiers
  Source    1..1412
            /organism="Citrus sinensis"
            /db_xref="taxon:2711"
            /clone="CitCHS2"
            /dev_stage="young seed"
            /note="Valencia orange"
  CDS       30..1205
            /codon_start=1
            /product="chalcone synthase"
            /protein_id="BAA81664.1"
            /db_xref="GI:5106369"
            /translation="MATVQEIRNAQRADGPATVLAIGTATPAHSVNQADYPDYYFRIT
            KSEHMTELKEKFKRMCDKSMIKKRYMYLTEEILKENPNMCAYMAPSLDARQDIVVVEV
```

```
                    PKLGKEAATKAIKEWGQPKSKITHLIFCTTSGVDMPGADYQLTKLIGLRPSVKRFMMY
                    QQGCFAGGTVLRLAKDLAENNKGARVLVVCSEITAVTFRGPADTHLDSLVGQALFGDG
                    AAAVIVGADPDTSVERPLYQLVSTSQTILPDSDGAIDGHLREVGLTFHLLKDVPGLIS
                    KNIEKSLSEAFAPLGISDWNSIFWIAHPGGPAILDQVESKLGLKGEKLKATRQVLSEY
                    GNMSSACVLFILDEMRKKSVEEAKATTGEGLDWGVLFGFGPGLTVETVVLHSVPIKA"
   polyA_site 1412
                    /note="18 a nucleotides"
BASE COUNT     331 a    358 c    372 g    351 t
ORIGIN
   1 aaacatattc attaagggtt caacttgaaa tggcaaccgt tcaagagatc agaaacgctc
   61 agcgtgccga cggcccggcc accgtcctcg ccatcggtac ggccacgcct gcccacagtg
   121 tcaaccaggc tgattatccc gactattact tcaggatcac aaagagcgag catatgacgg
   [...]
   1261 cacagttgag ttattggttg atcgtgtgaa ggtttagttt tgtcaattga gtttaaggca
   1321 tcgtgccttt tctcttatga cgtcaccaaa cctgggcaac gctttgtgtt tatgcataaa
   1381 ttcttgggaa tttgagaaag tagtaaattt gt
//
```

这个样品 GenBank 记录显示了可以在这个 GenBank 核酸数据库记录中找到的字段的类型。从蛋白质产物的鉴定（在本例中为苯基苯乙烯酮合成酶，chalcone synthase）、蛋白质产物的序列、基因中的开始和结束位点、提交记录的作者到描述实验的期刊文献等等任何事物，都可以在记录中找到，因此可以应用于数据库的搜索。

GenBank搜索界面与 PubMed搜索界面基本相同。在搜索工作中，Limits、Preview/Index、History 和剪贴板的特性与在 PubMed 的蛋白质、核酸和基因组数据库中相同，尽管可搜索特殊字段和可设置的限制有所不同。

保存搜索结果

序列可以三种文件格式中的任何一种从 NCBI下载：简单的 FASTA 格式，可被许多序列分析程序所读取，但除序列外仅包含极少的其他信息； GenBank的 flat 文件格式是在GenBank 中早期应用的 flat文件格式的遗迹；现代的 ASN.1(Abstract Syntax Notation One）格式。ASN.1是一个通用的数据规范，设计目的是用于促进数据库的协同工作的能力，现在用于 NCBI 所有数据类型的存储和检索，包括序列、基因组、结构和文献等数据类型。根据 ASN.1 规范，NCBI 的工具箱是开发分子生物学软件的代码库。NCBI 及正在发展壮大的其他组织，其软件开发都依赖于 NCBI 的工具箱。学会使用 NCBI 工具箱超出了本书讨论的范围，但是在 Web 上有由 Samuel Lunenfeld 研究所的 Christopher Hogue 及其研究团体开发的很好的自学教程。

除非数据库文件要被其他类型的软件使用或者阅读，熟悉的数据库用户或储存者不必考虑太多文件格式的问题。NCBI 的基于表单的界面将用户输入数据转换为恰当的格式存储，GenBank 文件的 FASTA 格式的可用意味着大多数序列分析软件可以不进行复杂的转换，而处理从 NCBI 下载的序列文件。

保存 GenBank 搜索的结果时，可以选择保存的格式。前面我们见过 GenBank 序列记录。我们在以后章节中要讨论的许多计算机程序可以阅读GenBank格式的序列文件，但是有些程序则不能。如果想用其他软件处理的话，最简单的用来保存序列文件的格式是 FASTA 格式。FASTA 文件具有一个简单的格式，首先是以 > 符号开始的一个简单的注释行，随后是保存一个序列所需的多行不间断的单字符 DNA 序列。当然，以 FASTA 形式保存数据时，有些与基因相关的信息会丢失，但是如果想使用的程序不能阅读那些额外的数据，无论如何它也是没有用的。

这里是一个 FASTA 格式的数据的例子：

```
> gene identifier and comments here
MATVQEIRNAQRADGPATVLAIGTATPAHSVNQADYPDYYFRITKSEHMTELKEKFKRMCDKSMIKKRYM
YLTEEILKENPNMCAYMAPSLDARQDIVVVEVPKLGKEAATKAIKEWGQPKSKITHLIFCTTSGVDMPGA
DYQLTKLIGLRPSVKRFMMYQQGCFAGGTVLRLAKDLAENNKGARVLVVCSEITAVTFRGPADTHLDSLV
GQALFGDGAAAVIVGADPDTSVERPLYQLVSTSQTILPDSDGAIDGHLREVGLTFHLLKDVPGLISKNIE
KSLSEAFAPLGISDWNSIFWIAHPGGPAILDQVESKLGLKGEKLKATRQVLSEYGNMSSACVLFILDEMR
KKSVEEAKATTGEGLDWGVLFGFGPGLTVETVVLHSVPIKA
```

为了以 FASTA 格式保存文件，仅仅使用结果页顶端的下拉菜单即可。第一次出现时是“Summary”，但是可以将它改变为 FASTA、ASN.1 及其他格式。一旦选择了自己想要用的格式，可以点击 Save 按钮保存所有序列到一个大的 FASTA 格式的文件中。图 6-5 显示了进行 GenBank 搜索时如何改变文件格式。

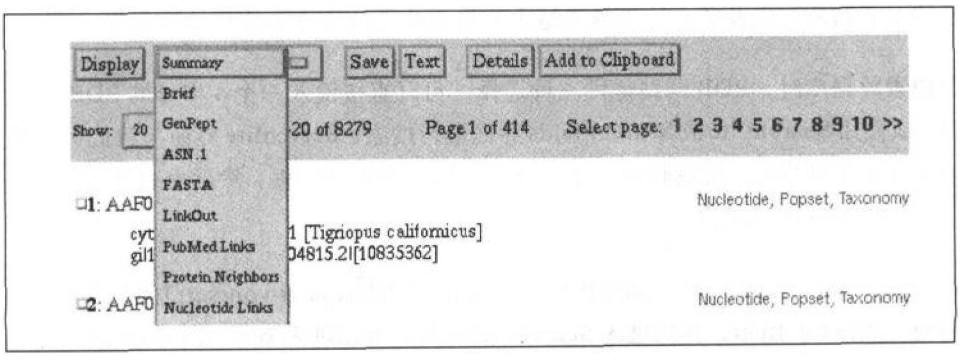


图 6-5　改变文件格式以写入 Genbank 搜索的结果

保存大的结果集

到目前为止，我们在讨论数据库的检索信息时，是假设只需要同时访问几个序列。然而，现代生物信息学研究处理大量序列数据的能力越来越强，比如，基因查找程序（在第七章中涉及）用成百上千的 DNA序列进行查找及检测；蛋白质家族的广泛研究也可能涉及上千个蛋白质序列的分析。尽管可以用手工在网页上选择上千个复选框，但是根据指定的标准用自动化工具返回大量的序列会更好一些。

NCBI在 Batch Entrez表单中提供了这样的工具（*http://www.ncbi.nlm.nih.gov/Entrez/batch.html*）。Batch Entrez是可从 Entrez 网站上得到的工具之一。可以使用一个 Web表单允许用户通过来源的生物体、 Entrez 查询（使用在有关 PubMed部分描述过的查询结构）或者通过编目列表（由用户以文本文件的形式提供）来选择序列。因此， Batch Entrez搜索结果可以打包到一个文件中，并下载到用户的计算机上，从而可以对完整的结果集人工或（甚至更好）使用脚本进行编辑。

目前，并不是所有的生物学数据库都可以提供比较好的服务，但是所有的公共数据库都具有 FTP站点，允许以这种形式或那种形式下载完整的数据库。它可能在硬盘上占据许多空间，但是这些磁盘空间的花费，与在一个交互式的 Web 站点上处理大量结果所花费的时间相比，要便宜得多。如果已得到一个感兴趣的大数据库的本地副本，你可以写（或可能甚至是下载）一个处理数据库的脚本，寻找你选择的关键词，写出在文件中想要的信息。

PDB

与 NCBI 不同，PDB（*http://www.rcsb.org/pdb/*）只负责一种类型的分子数据：分子的分子结构，以及所占比例越来越大的建立分子结构模型的原始数据集。

在搜索数据库时，PDB 网站提供三种选择：可以直接键入一个 4 字母的 PDB 标识符，或使用 SearchLite 或 SearchFields界面进行搜索。Searchlite 界面与我们讨论的其他查询工具相似，可以输入一个或者多个短语进入查询框，然后由运算符 AND、OR 和 BUTNOT来组合。

SearchFields 界面是一种有创造性的自己设计式（design-it-yourself）的 Web表单系统。如图 6-6 所示，首次进入 SearchFields时，可以进入 Web表单的底端，并选择符合需要的部分。如果只需要做一个 FASTA 搜索以找到相似的序列，那么不需

要提示使用在 Citation Author字段搜索中的关键词的搜索表单。你可能想加入一个字段，以允许用一个特定的配体或修复的基因来进行蛋白质搜索。使用 SearchFields 界面，你可以选择符合定制 PDB 搜索形式的元素，并点击“New Form”按钮以产生新的查询表单。

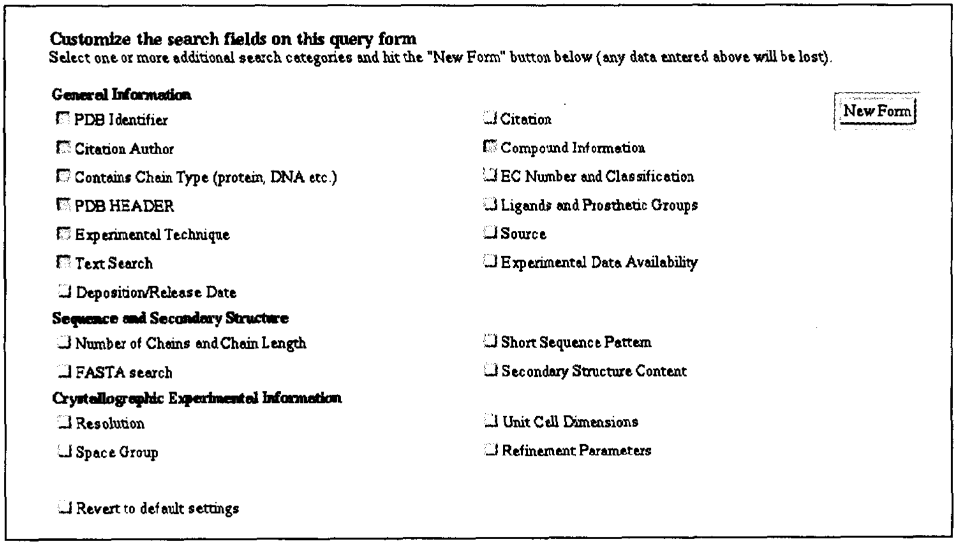


图 6-6　定制 PDB的 SearchFields格式

无论是用 SearchLite 还是 SearchFields，都可能得到 Query Result浏览器（见图 6-7），可选择改进查询的选项，下载作为结构或序列文件的结果，甚至制备出搜索结果的表格式的报告。这些选项可直接使用且在 PDB 站点中有很好的文档记载。

PDB 中的数据可以通过两种格式来获得，即传统的 PDB flat 文件格式或者新的 mmCIF 文件格式。我们将在第十二章中详细讨论这两种文件格式之间的不同。现在，可处理 mmCIF格式文件的结构分析和蛋白质建模软件很少（注 2），因此如果不是正在开发新软件，就不需要下载 mmCIF格式的蛋白质结构数据。可以从搜索中选择下载完整的结果部分，比如以 PDB 或 mmCIF 格式存为 *tar* 文档或 zip 文件，也可以以只有序列的 FASTA 格式进行下载。

注 2:　PDB 提供了一组 mmCIF 和 PDB 格式转换工具，以及处理 mmCIF 文件的代码库。

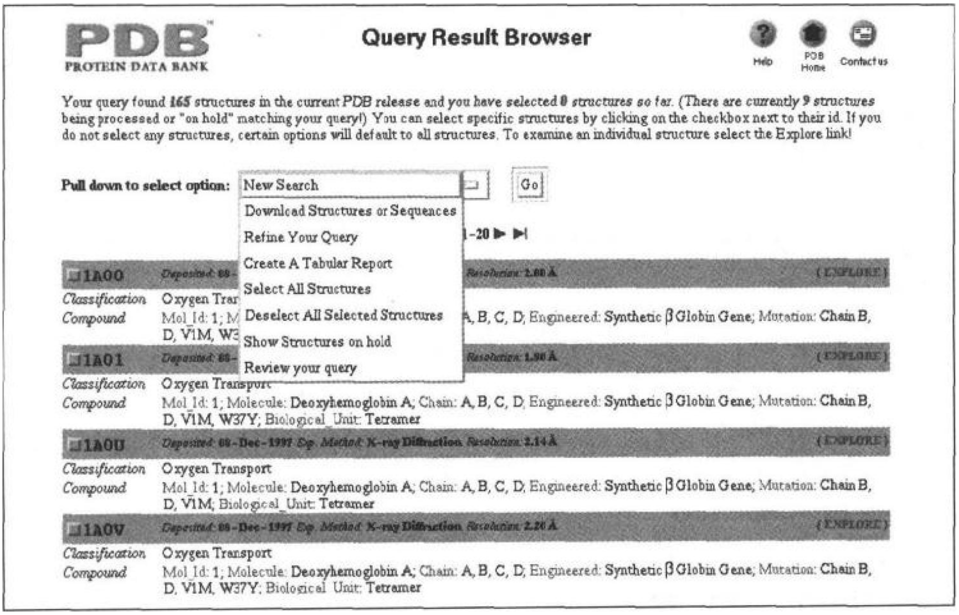


图 6-7　在 PDB 中应用查询结果的选择

另外一种从 PDB 站点查看蛋白质结构数据的方式是在计算机上安装一个浏览器插件，如 RasMol 或 Chime。我们将在第九章中介绍如何安装。一旦浏览器插件安装且恰当配置后，就可以直接点击蛋白质的 View Structure 页上的链接，蛋白质结构将使用浏览器插件自动显示，如图 6-8 所示。

在公共的数据库中存储数据

除了可从公共的数据库中下载信息之外，还可以提交自己的结果。

GenBank 存储

NCBI 使得将序列存储进 GenBank 已经变得非常简单。如果用户只储存少量的序列，可使用一个以 Web 为基础的 BankIt 工具，这个工具可以从 NCBI 的 GenBank 主页中获得，是一个可自我解释的、以表单为基础的界面。如果需要提交多个序列或者进行其他复杂的提交，可以使用 NCBI 的 Sequin 软件，这种软件在各种主要的操作

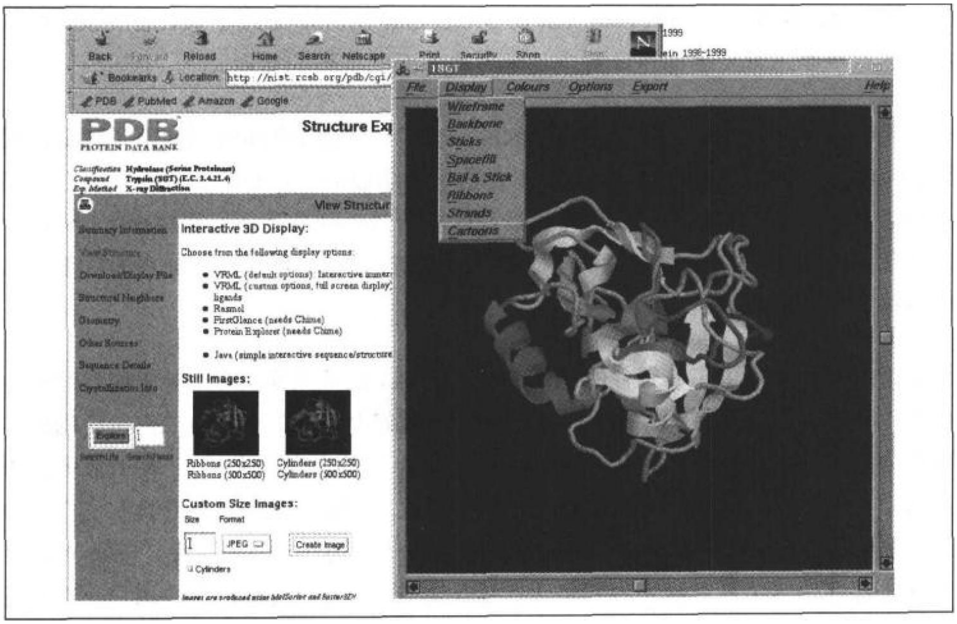


图 6-8　使用浏览器插件查看 PDB 文件

系统中都可使用。Sequin 在 NCBI 站点上有很好的文档记载。 NCBI 最近已经建立两个特殊的提交路径：EST 序列通过 dbEST 而不是通过 GenBank 提交，基因组序列可通过 dbGSS 提交。

PDB 存储

将结构存储至 PDB，可使用 AutoDep 输入工具（ADIT）来完成。AutoDep 是一个工具，它在存储过程中整合数据验证软件，以使用户可以在存储过程中收到数据质量的反馈。AutoDep 与在 PDB 中用来制备包含在数据银行中的结构数据的辅助工具绑定在一起。

查找软件

生物信息学是一个充满魅力的领域，吸引着来自多个领域的人才，有关生物信息学研究新进展的文章非常多。如果要查找尖端的发展，可查阅《Bioinformatics》、

《Nucleic Acids Researeh》、《Journal of Molecular Biology》以及《Protein Science》杂志，这些杂志常发表描述计算生物学方法方面的革新的文章。

如果要寻找已经得到证实的特定应用软件，有许多可靠的Web资源列出了可链接到计算生物学软件站点的列表。大多数主要的生物学数据库都有软件资源列表以及保持列表更新的必要动力。PDB链接到最好的免费的、用于大分子结构改良、可视化以及代谢的软件包。TIGR 和 NCBI提供了用于蛋白质和 DNA序列分析的许多工具的链接。

许多组织和团体提供了其软件的 Web实现。尤其是在应用研究中，对于使用非商业化的软件包的新手而言，这种方式是节约时间的好办法。在本书中我们讨论的许多生物信息学的程序也可以作为 Web 服务器获取。可以使用该程序的 Web 服务器版本入门，来理解该程序的输入、输出和各种选项。然而， Web服务器具有它们的缺点。一般而言，它们只能实现上述任何软件包中的最通用的选择，设计一种Web 表单以允许选择一个复杂程序中的每一选项是困难的。它们通常只允许用户一次进行一个计算。如果只对分析很少的序列或结构感兴趣，那么没什么问题；但是，如果你突然发现自己需要分析 500 个序列，那就有点麻烦了。

采用更智能一些的编程方法，可以开发允许使用多次请求来访问一个 Web服务器的脚本，而不必手工将这些请求输入表单，但是，如果你能够做到这一点，就很可能可以下载一个软件的本地副本，并在自己的机器上运行该软件。在这种情况下，运用自己的处理器进行处理，可以避免数据缓慢地从远程站点传入或者传出到远程的站点；并且，相对于在其他人的 Web服务器上进行大量的工作，下载并在本地运行这些软件要显得更有礼貌一些。

在接下面的四章中，我们将讨论最可能用到的软件包。我们将介绍怎样在自己的计算机上安装这些软件包，并且介绍如何不依赖 Web界面使用它们。

我们不可能在本书中覆盖所有可获得的软件包及Web服务器，因为它们实在是太多了。最终，大家可能会自己寻找一些新的软件工具来使用。在自己寻找软件时，如果稍加注意的话，应该很快就能判断一个新的计算机程序是不是自己想使用的程序。

判断信息的质量

如果不断地在此领域学习，那么你判断从Web上查找到的信息和软件的质量的能力将会提高。然而，非常明显，一些简单的指导可以帮助你筛选所查到的信息。对Web上提供的软件、信息以及服务使用强健的怀疑主义，也就不会轻易地误入歧途。

权威性

对因特网上发现的软件、数据或信息进行评估时，首先要考虑的问题之一是其来源。其作者是谁？如果你不能通过其声誉知道发布信息的作者信息，是否可以通过网站来获得有关他们的附属机构和可信度的信息？他们的专业是否和网站的主题或目标相关？对你来说，是否有可能和他们联系并向他们提一些问题？

发起信息的组织的目的是什么？它是一个学术机构、一个政府组织，还是一个公司？赢利的公司与非赢利的组织和学术研究团体相比，在提供可用的软件方面经常具有不同的目的；通常他们会提供一个有限制的软件版本或服务来促使你购买更完整的软件包。个人的学术研究站点不像一般的公共投资的数据库那样无所不包，提供的这些东西没有本质上的问题，但是你应该明白它们是否全面，它们的内容是否适合于一般的用户，以及为什么会适合一般的用户。

即使是来自国家或国际的公共站点的数据或者软件，可能都不是完全正确的。有人估计在GenBank中，任何给定的序列都可能包含至少一处的错误。在了解到这些错误一般不会导致数据无意义的同时，最好能明白即使是使用最好的公共资源也会存在这样的问题。就像在Web上发现的其他软件一样，由公共机构比如NCBI以及PDB提供的软件也在开发过程中。可以使用这些软件，其中大部分的质量都不错。如果根据一个软件包的β-版本（还在开发过程中的版本）进行自己的研究，要仔细阅读说明，看还有什么需要解决的问题。

透明度

发送数据给Web服务器去处理时，你是否确切地想到将会发生什么事？你应该想一下。运用文字处理器做一个黑盒不错，但是如果要发表的科学结论是基于Web服务器的软件包，应该明确地知道其基础是什么。根据任何的软件，不管它是好的还是

差的，任何人都可创建一个Web 服务器。创建一个 Web服务器，也就创立了一个关于权威的幻觉；毕竟，作者知道怎样创立一个运行的 Web服务器，因此，他们的其他软件也可运行。但是权威的形象经常不是建立得很好。

理想情况下，可得到 Web服务器如何运行的源代码（一个计算机程序的人类可读的版本），你可以读懂源代码，并且知道它如何按期望的去工作。但是如果你不知道如何阅读源代码，即使这样做了，也不能明白它的含义。不幸的是，一些生物信息学软件的作者不会公开他们的源代码，而且倾向于使 Web服务器简单易用、易于保持。这些可能会向详细审查的用户隐藏底层的方法。

如果不能读懂源代码，那么你能读懂什么？许多由学术研究者或政府研究所开发的软件或Web服务器有在线帮助和其他文档说明，包括在相关的期刊中发表描述软件编码的方法的参考文献信息。在使用之前，可以阅读这些文档，理解这些方法和结果，就像面对一个新的实验方法时要做的一样。

如果想使用的程序或服务器既没有文档，说明又不允许检查源代码，如果不能用其他一些方式来证实它的输出（比如，通过与一个已得到广泛证明的程序的输出进行比较），应该慎重地考虑不使用这个程序。毕竟，我们要根据结果来下结论，怎么能够用未知数来保证科学可信性呢？

时间性

一个最常被链接的 Web上的生物学资源站点是 Pedro 的 Biomolecular Research Tools（生物分子研究工具，*http://www.beri.co.jp/Pedro/research_tool.html*）。世界范围内的站点都有指向这个链接的指针。然而，如果点击 Pedro的站点，你将会发现其最后的更新是在 1996年。一个可笑的有关 Web的事情是过时的站点并不消失。它们保存在服务器上，看起来具有权威性。检查Web站点的时间。如果没有活动性的标记或者当年的文献，就可以忽略它们。

软件通常没有时间性问题。1980年写的软件，现在可能仍是有用的。我们可能遇到的问题是属于机构专利技术的编译软件不再被支持，或者是代码库已经停止开发。

第七章

序列分析、成对比对以及数据库搜索

现在，我们以极大的热情开始生物信息学工具学习的旅程。在以下 5 章中，我们将介绍在计算生物学领域的研究中，大家希望学习到的软件工具及应用程序。从基因序列、基因编码的蛋白质到它们参与的复杂的生物学网络，计算工具都可以用来分析数据和拟订假说。我们现在要集中研究常用软件包以及我们已经用过的软件包；然而试图包括程序的每个细节，需要把本书的每一章都变为独立的书籍。

我们介绍的第一个工具是分析蛋白质和DNA 序列数据的软件包。序列数据是可获得的电子版生物学数据中最丰富的一种。虽然其他类型的数据库最终可能在大小上可与序列数据库进行抗衡，但是序列数据库对生物学来说依然是最重要的。我们在本章中介绍的成对序列比较，是计算生物学中最重要的技术。它允许我们做包括以序列为基础的数据库搜索、建立进化树、鉴定蛋白质家族特征性的性质和建立同源性模型在内的所有工作。然而，它也是非常大的课题的关键 —— 基因组比较、探索蛋白质结构的序列决定因素、将表达数据与基因组信息联系在一起，等等。

可对序列数据进行分析的类型有：

- 对以知识为基础的序列性质的单一序列分析
- 成对序列比较和以序列为基础的搜索
- 多序列比对

- 在多序列比对中发现序列基序
- 系统发育的推论

我们将本书所覆盖的有关序列分析工具的内容分为两章。本章的内容集中在对单个序列进行操作的程序，或者基因或蛋白质序列之间的比较。第八章将主要讲述多序列的比对方法。

成对序列比较是将生物学功能与基因组相联系，以及将已知信息从一个基因组推广到另一个基因组的主要手段。在本章中，我们将讨论生物学序列分析的技术，更重要的内容是怎样评估从序列比较中得到的结果的意义。这里还将介绍有关成对序列比较的许多可获得的软件工具。表 7-1 列出了这些工具。

表 7-1　序列分析工具和技术

目的	原因	可用工具
查找基因	在基因组 DNA 序列中鉴定可能的编码片段	GENSCAN,GeneWise,PROCRUSTES，GRAIL
DNA 性质的检测	剪接位点、启动子、参与基因表达的调节序列的确定	CBS Prediction Server
DNA 翻译和逆翻译	将 DNA 序列转化为蛋白质序列或者其逆过程	在 EBI 上的“Protein machine”服务器
成对序列的比对（局部）	在一对长序列中的短片段的同源性确定	BLAST，FASTA
成对序列的比对（全长）	在两个序列之间找到最好的全长的比对	ALIGN
成对比较的序列数据库搜索	查找关键词搜索时不能识别的序列匹配；查找实际上具有序列同源性的匹配	BLAST，FASTA，SSEARCH

生物分子的化学成分

序列分析技术可用于 DNA 和 RNA （核苷酸）序列或者蛋白质（氨基酸）序列。为了理解为什么 DNA 及蛋白质序列可以提供信息，了解有关 DNA 和蛋白质化学性质

的知识是必要的。在本章中，在讨论序列分析应用程序的内容时，最好将DNA序列看作单纯的信息。

可以忽略化学结构，认为DNA是一个字符串。但是在脑海中应该很清楚，描述DNA的单个字母的序列代码是一个三维化学实体的简化，在有些情况下，DNA的三维结构的确很重要。

至少一眼看上去，蛋白质比DNA的化学结构更复杂，要将其序列的信息内容与构建蛋白质的氨基酸的化学特性分开是不可能的。进行序列分析时，不能完全忘记蛋白质的化学结构，因此，在第九章的开始，在介绍蛋白质结构分析技术之前，我们将大略地讨论蛋白质的化学性质。像在第二章中讨论过的一样，DNA是细胞中存储信息的介质，它通过组成 DNA 链的核苷酸序列存储和传递信息。 DNA 是“双螺旋”结构，即两个长核苷酸序列，从化学结构来讲，这两个序列互为镜像。这种双螺旋结构和螺旋中两部分核苷酸的特殊配对模式的化学性质，赋予了 DNA 自我自制的能力，以及从细胞到细胞、从父代到子代忠实地传递信息的能力。核苷酸链之间配对的化学性质也允许 DNA 序列被转录为 RNA 并翻译为蛋白质。

DNA和 RNA 的组成

DNA 和 RNA 是由小写字母拼写而成的具有相似化学结构的多聚体链组成的。每一个单元称为一个核苷酸，如图 7-1所示，核苷酸由三个不同的部分组成：一个环状的碱基、一个环状的糖（分别为脱氧核糖或核糖），以及一个磷酸基因。在 DNA和 RNA 中对碱基的利用是不同的。 DNA由A(腺嘌呤)、T(胸腺嘧啶)、G(鸟嘌呤)、C （胞嘧啶）核苷酸编码而建立的模式组成。而在RNA 的编码中 U(尿嘧啶)代替了 T。

在图 7-2中显示了5种核苷酸，它们也用来表示碱基。在水解的双链 DNA中，通常是 A和T的数量相等(A=T)。G 和 C 的数量相等（ G=C)。在研究人员发现 A 和 T、G和 C之间的关系之后，这种关系被命名为 Chargaff 规则。（注意 A和 T与G 和 C 的数量可以有不同。在不同的物种之间， A-T和 G-C碱基对的比值可以有明显的不同。）

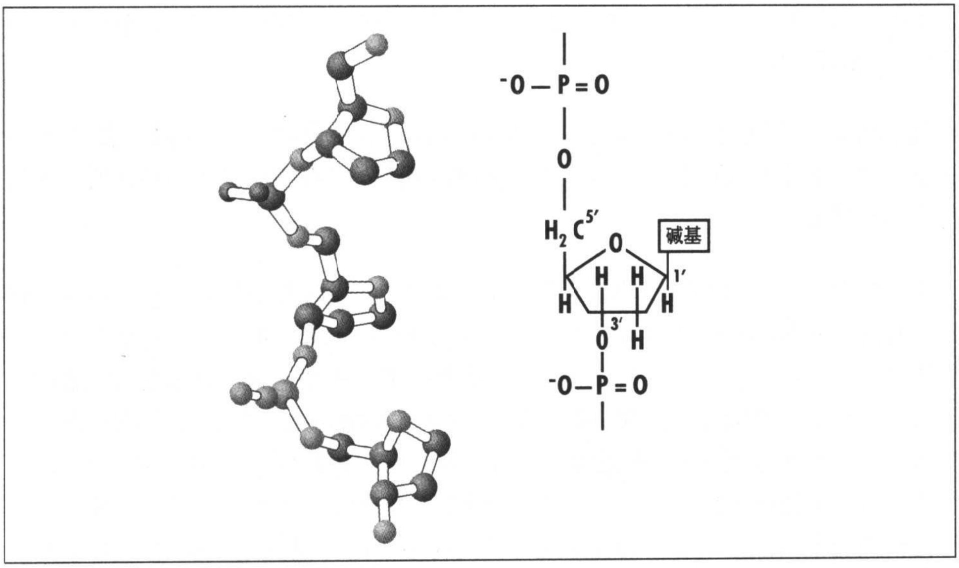


图 7-1 DNA 和 RNA 的‘骨架’单位——核糖和脱氧核糖磷酸

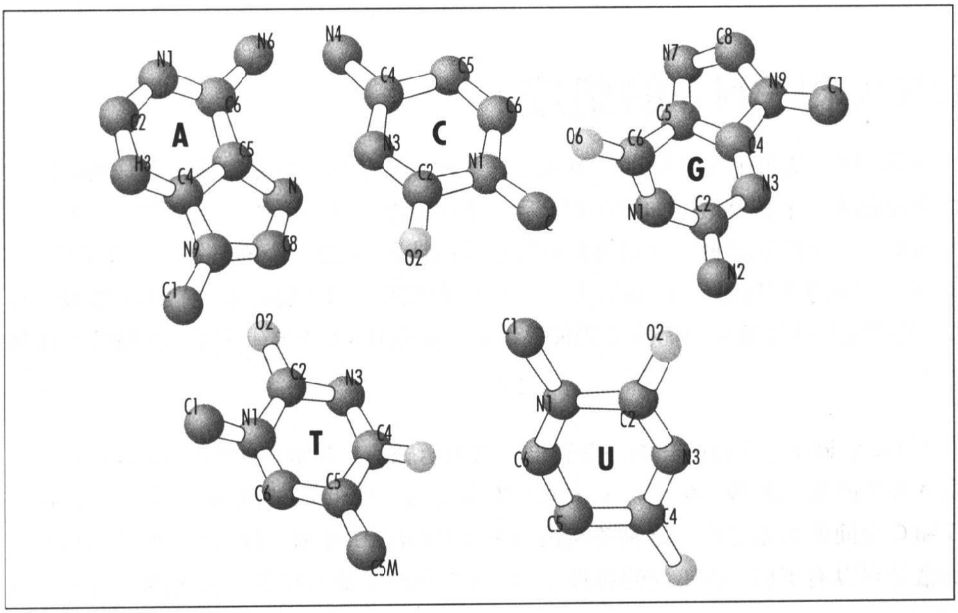


图 7-2 一般在 DNA 或 RNA 中出现的五种碱基

Watson及Crick解决了DNA 的结构问题

A和T、C 和 G之间的数量关系使得 Watson及 Crick在 1953 年建立了DNA 的结构模型以及后来的 Crick的生物学中心法则。Watson及 Crick的 DNA 模型是建立在以下基础上的：

Watson及 Crick的同事Rosalind Franklin 从X射线结晶学实验中发现 DNA的衍射模式,提示 DNA是一个双螺旋分子，具有规则的重复结构，间隔为3.4埃。

Chargaff 规则。

实验证明，DNA 分子的碱基是由氢键连接的。

从 X 射线结晶学中得到有关碱基正确的结构构象的知识。

Watson及Crick所做的工作是将分开的信息结合到一起，提出了双螺旋结构。有关DNA双螺旋结构原子的细节现在可用 X射线结晶学确定，DNA双螺旋结构是一个通过氢键 A与 T配对、G与C配对的结构（如图 7-3)。氢键结合的碱基对形成分子的中心。

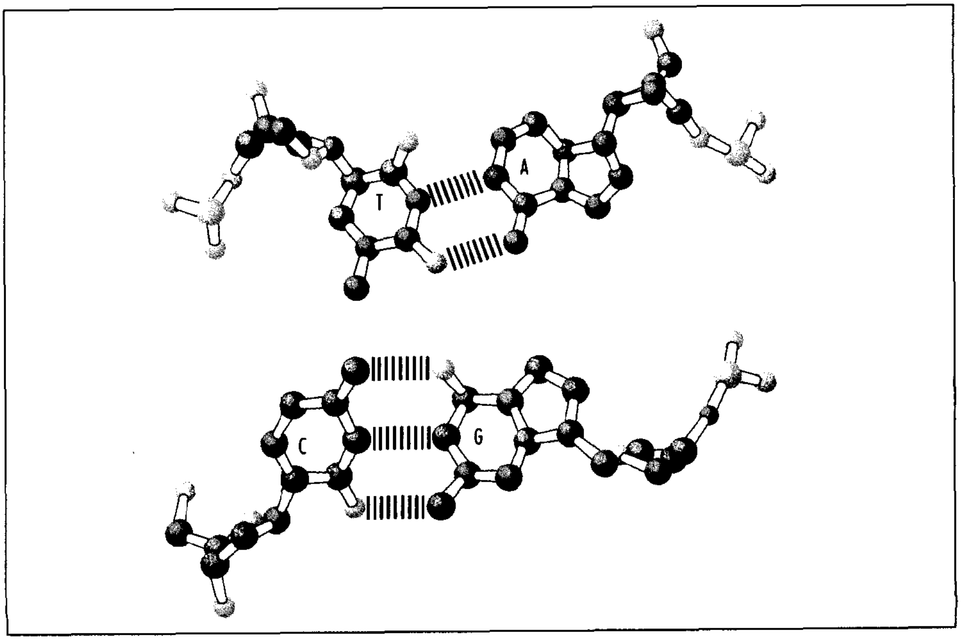


图 7-3　两个常见的碱基对，A-T 和 G-C

如图 7-4 所示，碱基对相互上下堆积，并且碱基对之间相互平行，间距为 3.4 埃 。在序列中每个核苷酸的糖基以及下一个核苷酸的磷酸基之间由共价化学键结合到一起。这个 DNA 链具有方向性：暴露磷酸基团的一端，被称为是 5' 末端；暴露核糖基的 DNA 链末端为 3' 末端。

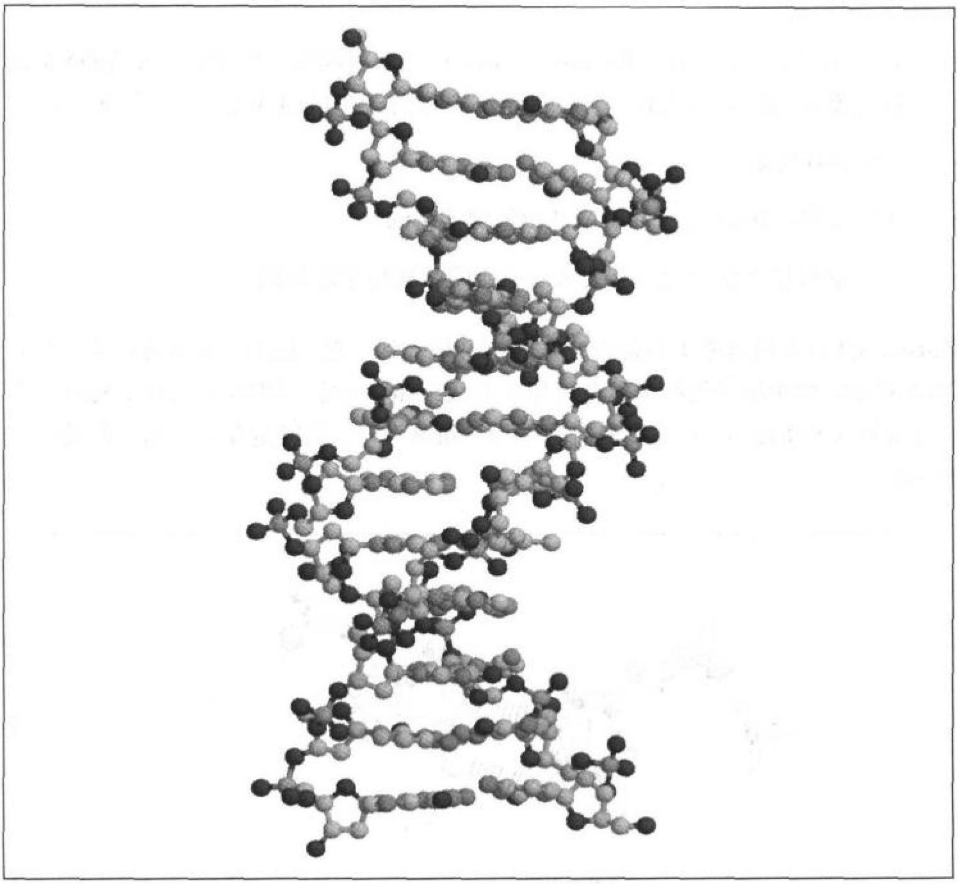


图 7-4　DNA 链简图

在 DNA 和 RNA 序列中，特定的核苷酸的化学配对形成一种特殊机制，使 DNA 的每条链都可以做模板来合成另一个互补链。在 RNA 中应用相似的核苷酸代码提示 DNA 可以作为模板合成 RNA。从上述两条证据中，Crick 得出他的中心法则：DNA 指导自身复制以及 DNA 转录为 RNA 的过程，RNA 可翻译为蛋白质。

DNA 测序方法的发展

如果把DNA 只理解为四种碱基，并检查它们各自的数量，那么你不能得到任何有关DNA序列的信息。当前对 DNA 进行测序的方法依赖于可控制的生物化学反应，DNA 序列每个位置的碱基含量可以独立的定量。 DNA 测序的化学裂解方法，依赖于裂解DNA 链中 4 种类型位点的化学反应试剂的专一性。有的试剂可以指定在 G之后打开或断裂 DNA 链，有的试剂可以在 C之后断裂 DNA链。也有一些试剂裂解时专一性不强：一种在 A 和 G之后裂解，另一种在 C和 T之后裂解。Maxam 和 Gilbert 设计的方法在概念上很简单，这种方法需要4个DNA 样品，一种类型的试剂定量与DNA 样品混合，使得样品中的每一个 DNA 链平均在一个位置只随机断裂一次。DNA 的一端用放射性标记，另一端不用标记，因此，在 DNA链断裂之后，每一断裂的链之后只有一段是具有放射性活性的。 DNA不同大小的片段、可以在称为凝胶的粘性介质中用电流促使它们分离。片段越大， DNA在凝胶中泳动得越慢，一段时间后就会越靠近凝胶的末端，不同大小的放射性活性DNA就会以规则的间隔分布在凝胶上。图 7-5显示一个 DNA 测序的凝胶的部分放射自显影图像。在空间上接近的四条带中，每一条带都代表一个独立的序列，凝胶是从底部向顶部阅读的，凝胶上的每一条带表明序列中核苷酸的位置，而其位置取决于它出现在四条带中的哪条带上。如果每条 DNA 链在 A、C、G、T 后面随机打开一次，那么表明 DNA 整体序列的片段分布就可以完全表现出来。根据放射性片段来源的样品，以及已知的序列的最后一个碱基，序列就可以从凝胶中从头至尾读出。

现代实验室中应用最多的测序方法是Sanger的链 – 终止子过程。这个过程的优势在于应用的是 DNA 多聚酶，有了这种酶，可以根据一条链来复制另一条链。在 Sanger 方法中，DNA 多聚酶反应的实现是由于序列中核苷酸特异性类似物的存在。当核苷酸特异性类似物作为原料进入到核苷酸链后，引起复制反应停止。四个样品准备好后，每一条链都包含少量的同一类型的终止子。类似于 Maxam 和 Gilbert 的方法，得到 DNA片段完整的分布后，每一条链均有一个可知的末端残基。根据荧光信号的强度可以分析片段，然后根据互补的 DNA 序列给出原来的 DNA 序列。

DNA 链的末端分析法很容易自动化，应用这种方法的计算机兼容测序系统很容易读出结果。大多数基因组序列数据是应用这种方法产生的，虽然正在开发的许多新的测序方法没有链的裂解或链末端的参与。在第十一章中我们将深入讨论测序的数据分析过程和基因组装配。

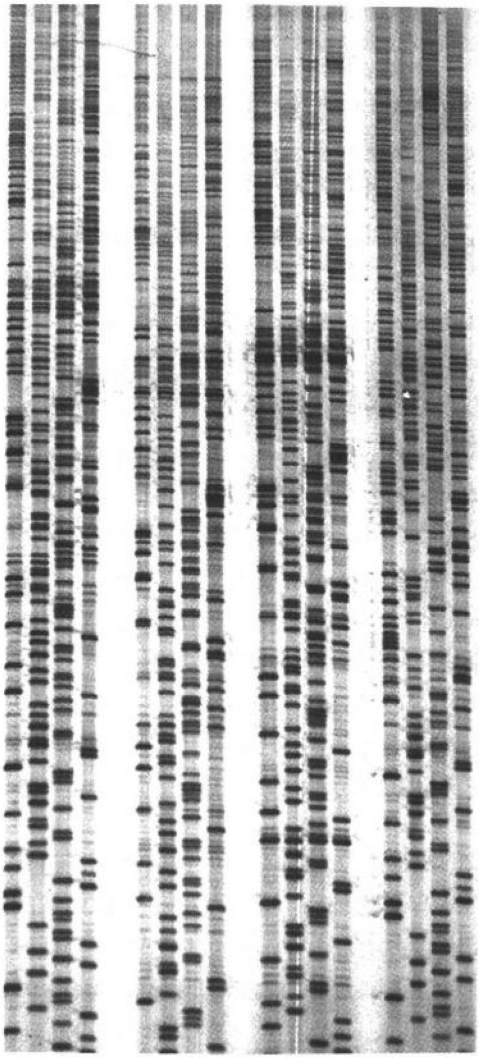


图 7-5　　DNA 测序凝胶

蛋白质的化学成分

和 DNA 不一样，蛋白质多聚体是由一系列常见的、称为氨基酸的分子组成的。有20种氨基酸组成标准的化学字母表用于构建蛋白质。氨基酸是共享常见基序的小分

子，在一个中心的碳原子周围有三种取代化学基团。其中的一种取代基团总是一个氨基基团；另一个基团总是羧酸基团。为了组成蛋白质多聚体，氨基与羧基相互反应并形成一个肽键，氨基酸的中心碳原子的第三个取代基团是可变的，这是将氨基酸组成存储信息的代码特性。蛋白质中的氨基酸序列指的是蛋白质的一级结构。蛋白质序列可以进行和 DNA 序列同样的分析（以后讨论）。在描述序列分析方法时，我们将指出这些方法中蛋白质和 DNA 的不同。

分子进化的机制

作为遗传和进化的分子基础的DNA的发现，使生物工作者以一种全新的方式理解进化的过程成为可能。达尔文的自然选择进化理论描述了可观测到的进化和物种形成的过程。然而，它不能解释信息怎样从一代传给下一代，也不能解释在每一代中变异是如何出现的以及变异的界限。

DNA 双螺旋分子的两条链可以作为复制的模板。即使控制 DNA 复制的分子法则是特有的，其复制过程也并不总是具有非常好的保真度。当一段 DNA 分子被不正确地复制，并且其错误不能被细胞修复系统正确地纠正时，就称之为突变（mutation）。

突变在机体 DNA 的任何部分都可能发生：在编码蛋白质或功能 RNA 分子的基因中间、在控制基因开关的调节区域、在“不为人知的地方中间”之外或者在基因序列之间的区域，突变可对有机体的表现型具有明显的作用（可以是看得见的或可检测到的特性），或者没有明显的变化。长时间 —— 在成千上万年以后，对种系有益的或至少无害的突变可在群体中固定下来，这意味着在一个特定的种系中，所有个体的基因突变具有一定的频率。长时间之后，大量的突变累积，可导致新的物种的形成。

存在两种类型的突变：点突变（point mutation）为影响 DNA 序列中一个核苷酸的改变；以及片段突变（segmental mutation），可以影响任何位置中从几个到几百个相邻的核苷酸。

点突变通常只导致一个单个的错配，在一个新的互补DNA 链中，一个核苷酸与模板DNA错配。只有当点突变发生在编码区域或者是信号序列，导致功能上的改变时，突变才会变得有意义。在编码区域，点突变可能是同义的，意味着对相同的氨基酸来说突变后的链编码和突变之前相同，点突变也可能是不同义的。遗传密码子（在

图 2-3 中示出）具有简并性，也就是说，每个氨基酸具有几个不同的 3个字母结合在一起的密码。编码每个氨基酸的密码子群绝不是随机的，自然界已安排了一个失效保护机制，只有一个核苷酸不同的几个密码子代表同一个氨基酸，因此给了同义突变的 DNA 复制错误的一点空间。

片段突变，可以导致长的 DNA片段的插入或缺失，可由许多不同的机制引起。所有机制都与 DNA链的错配有关，可以是互补链的错误或者是它自身的一部分错误。片段突变会导致整个基因的重复或甚至是染色体的大的片段的重复；一些遗传事件就是由于整个基因组的重复导致的。由基因及染色体重复导致的，基因的多余的副本可以被再利用（通过突变的试验和错误的缓慢的过程）形成细胞的新功能。有关这些机制的详细讨论在一本很好的书《Fundamentals of Molecular Evolution》中有介绍，可见参考文献。

两种类型的突变在进化记录中，即生物的 DNA 序列中留下了线索。虽然在功能上是有益的（至少是无害的）突变倾向于被保留，但是功能上较重要部分的序列倾向于保守（在进化过程中保持不变），而不编码或无功能的序列可以有非常大的差异。功能比较重要的序列比较保守，这种倾向是整个序列分析的基础；它可使我们抓住与序列相关的基因之间进化上的联系。

通过DNA 序列以及大范围内乃至整个基因组的比较研究，为理解突变事件何时以及怎样发生，以及突变在生存的种系和群体中如何及为什么能够保留而建立定量的方法是可能的。基因组学和生物信息学（基因组数据的产生以及数据分析工具的形成）已经使检测进化记录以及建立越来越多地定量描述一个种系到另一个种系之间的进化关系成为可能。分类学开始不仅以解剖学为基础，而且以遗传密码中差异定量测定为基础。点突变和片段突变在本章后面讨论的蛋白质和DNA序列的对比中以记分法的方式明确地建模。序列中一个给定位置的残基（核苷酸或氨基酸）的一致变化可用标准的替代记分法进行记分（比如，正分表示匹配，负分表示不匹配）或者应用替代矩阵法，对空位开放和空位延伸中的插入和缺失罚分法进行记分。

genefinder 和 DNA 特征的检测

在大量的 DNA图谱以及测序完成之后，理解其功能的任务就开始了。本节将描述在序列中搜索基因以及其他生物学重要特征的程序。特征（ feature）是指具有功能意

义的序列模式。比如起始和终止密码子、剪接位点（在真核细胞中），以及为调节基因表达而被蛋白质结合的序列。一些特征可以通过搜索一个特定的序列而发现，比如被一个给定的限制酶切开的酶切位点。其他的特征，比如启动子和基因，则不太容易被发现。在序列特征的搜索中单一DNA序列的分析是生物信息学中快速发展的研究领域。

有两个原因使基因查找及特征发现成为著名的难题。首先，有大量的蛋白质 -DNA 相互作用，而且还有许多未被实验所描述，而且有些蛋白质 -DNA 的相互作用在不同的有机体中也有差别。更重要的是，我们并不总是很清楚是什么构成一个结合序列。现在启动子检测算法对于每一个真正的启动子的确定来说，会产生 20 ~ 40个错误的阳性结果。有一些蛋白质结合到特异的序列，而其他的蛋白质在结合位点的优先选择中具有更多的弹性。更加复杂的是，蛋白质可以只结合到染色体的一部分，但却会影响到成百上千的碱基对以外的完全不同的区域。

预测基因定位

genefinder是鉴定（或试图鉴定）未注释的 DNA 中所有开放阅读框的程序。它们使用一系列定位基因的方法，但是最成功的方法是将以内容为基础的方法与模式识别相结合。以内容为基础的基因预测的方法利用基因中核苷酸的分布与非基因序列不同的事实。由 Oak Ridge国家实验室开发的程序 GRAIL家族使用神经网络结合 DNA 内容的七个不同的统计测量（框偏爱、周期、不规则碎片形的维数、编码的 6-元组、框内的 6-元组，k- 公共元组和重复的 6-元组命令）的证据；随后的版本测量附加特征可以更好地利用这些不同类型的数据。在DNA序列的每一个位置，程序将权衡每种类型的信息，对它们进行整合，并计算出一个分值，代表所查询的区域是在 ORF 中，还是在基因间区域的可能性。模式识别方法寻找和基因相关的特征性序列（比如起始和终止密码子、启动子、剪接位点）以推测基因的存在和结构。

如果每个方法分开使用，效果会很一般。如果通过分别寻找特征性的皮肤质地（内容）或是胡子（模式）的存在来鉴别人脸，而不是同时使用，成功率与此类似。因此，现在 genefinder的形成结合了两种方法以及附加的知识，比如基因结构或其他的已知基因的序列，这一点并不奇怪。

一些genefinder只能通过 Web界面进入，使得其相互作用非常直接：需要检测基因的序列输入到程序中，被处理之后便得到输出结果。一方面，这样可以不用在自己

的系统中安装和维护 genefinder，对于不同的程序它会提供一个相对统一的界面。另一方面，如果要依靠 genefinder 的结果做下一步的工作，应该花时间理解其底层算法，查出模型对一个给定的种系或家族是否是特异的，对于以内容为基础的模型来说，应该知道它们是哪一些序列。如果用同样的序列去检测，就会发现 genefinder 的精确度会有非常高的误解率。

在查找基因时，一些常用的程序有 Oak Ridge 国家实验室的 GRAIL，GENSCAN（由 Chris Burge 和 Samul Karlin 开发，现在 Chris Burge 在 MIT 工作，Samul Karlin 在 Stanford 工作），PROCRUSTES（由 Pavel Pevzner 及其同事开发），GeneWise（由 Ewan Birney 及 Richard Durbin 开发）。GRAIL 使用神经网络结合了一系列信号及内容的信息。GENSCAN 将有关内容统计学的信息与基因结构的概率结合在一起。PROCRUSTES 及 GeneWise 通过翻译 DNA 序列，并把得到的结果蛋白质序列与已知的蛋白质序列相比较，来发现开放阅读框。PROCRUSTES 将潜在的 ORF 与同系相似物进行比较，而 GeneWise 将基因与整个蛋白质家族的序列或模型进行比较。

特征检测

特征检测算法除了在 genefinder 系统中发挥作用外，也可在 DNA 序列中用于查找模式。这些工具常用于帮助解释新测序的 DNA、为设计 PCR 引物和微阵列寡聚体选择靶点。几个工具的发起中心包括 Denmark Technical 大学的生物学序列分析中心（有几个基于 Web 的、可用来查找真核细胞 DNA 的内含子－外显子剪接位点和转录起始位点的应用程序），Fred Hutchinson 的癌症研究中心的 CodeHop 服务器（根据保守的蛋白质序列可以预测 PCR 引物）以及欧洲生物信息学研究所的 Tools 集合。

除了这些具有特定目的的工具之外，另外的常用方法是使用可自动查找序列的常见模式的 motif 发现程序。在介绍多序列分析方法时，我们将更仔细地检测这些程序。

DNA 翻译

在蛋白质被合成之前，它的序列必须从 DNA 中翻译出来。将 DNA 序列翻译为蛋白质序列在概念或者计算上并不困难。需要的是 DNA 序列、遗传密码能够读取一种类型并输出另一种类型的序列的程序。

任何 DNA序列都可以以 6种可能的方式进行翻译。序列可以向前或向后翻译。因为蛋白质中的每个氨基酸都由 DNA 序列中的三个碱基指定，任何 DNA 在每一方向都有三种可能的翻译方式：从序列的第一个字母开始、从第二个字母开始，以及从第三个字母开始。

图 7-6 显示的是使用细菌和植物质体遗传代码，蛋白质序列被“逆翻译（back-translation）”成 DNA 。如我们所看到的一样，从蛋白质序列“逆翻译”成的 DNA 并不是惟一的。显示的短序列中每个氨基酸可以由 6 个代码来代替，即使是一个短的肽，可能的密码子也可以通过多种结合，产生上百种可能的编码序列。然而，自然已经为密码子进行了有意义的分组。丙氨酸（alanine，A）通常为“G-C-X”密码子所指定，精氨酸(arginine，R）是由‘C-G-X”密码子或者是‘A-G-pyrimidine”密码子所指定。如果（比如）想写一个程序将蛋白质序列与 DNA序列数据库进行比较，这种自然选择就减少了必须检查的可能序列的数目（注 1)。

A	L	Q	E	R	T	A	S	P	G	S	G	蛋白质序列
GCC	CTT	CAC	GAA	CGT	ACT	GCC	TCT	CCT	GGT	TCT	GGT	可能的密码子
GCT	CTC	CAA	GAG	CGC	ACC	GCT	TCA	CCC	GGC	TCA	GGC	
GCA	CTA			CGA	ACA	GCA	TCC	CCA	GGA	TCC	GGA	
GCG	CTG			CGG	ACG	GCG	TCG	CCG	GGG	TCG	GGG	
	TTA			AGA			AGT			AGT		
	TTG			AGG			AGC			AGC		

图7-6　蛋白质序列的逆翻译

在DNA 序列中没有标记来提示一个密码子结束，下一个密码子开始。因此，如果起始密码子的位置不是预先已知的话，一个双螺旋DNA序列可以用 6种方式的任何一种来解释：在所观察的链或者是在互补链中，开放阅读框可以以核苷酸： *i*、*i*+1 或 *i*+2 开始。为说明这种不确定性，当一个蛋白质与一组序列相比较时， DNA序列被翻译成所有6种可能的氨基酸序列，蛋白质查询序列与这些产生的翻译物进行比较，这种详尽的翻译称为“6-框翻译”，如图 7-7 所示。

注 1：　此问题更高效的解决方法是在所有 6 个阅读框中翻译 DNA序列数据库。

```
>gi | 9858881 | gb | AF 291052  Zea  mays  subsp.  parviglumis  hemoglobin gene
DNA :  ATGGCACTCGCGGAGGCCGACGACGGCGCGGTGGTCTTCGGCGAGGAGCAG
 +3 :    G  T  R  G  G  R  R  R  R  G  G  L  R  R  G  A  G
 +2 :   W  H  S  R  R  P  T  T  A  R  W  S  S  A  R  S  R
 +1 :  M  A  L  A  E  A  D  D  G  A  V  V  F  G  E  E  Q

DNA :  GAGGCGCTGGTGCTCAAGTCGTGGGCCGTCATGAAGAAGGACGCCGCCAAC
 +3 :    G  A  G  A  Q  V  V  G  R  H  E  E  G  R  R  Q  P
 +2 :   R  R  W  C  S  S  R  G  P  S  *  R  R  T  P  P  T
 +1 :  E  A  L  V  L  K  S  W  A  V  M  K  K  D  A  A  N

DNA :  CTGGGCCTCCGCTTCTTCCTCAAGTAAGTACGTTTCCGTGCTACACACTGC
 +3 :    G  P  P  L  L  P  Q  V  S  T  F  P  C  Y  T  L  P
 +2 :   W  A  S  A  S  S  S  S  K  Y  V  S  V  L  H  T  A
 +1 :  L  G  L  R  F  F  L  K  *  V  R  F  R  A  T  H  C

DNA :  CTGCGCACGTGCGCTTGGGTTGCACCTGCACCGGCGGCCATCGAGCCTGCT
 +3 :    A  H  V  R  L  G  C  T  C  T  G  G  H  R  A  C  S
 +2 :   C  A  R  A  L  G  L  H  L  H  R  R  P  S  S  L  L
 +1 :  L  R  T  C  A  W  V  A  P  A  P  A  A  I  E  P  A
```

图 7-7　DNA序列和六种可能的阅读框中的三种翻译产物

由于一些氨基酸存在大量的密码子的可能性，蛋白质“逆翻译”为DNA序列导致可能的序列数目非常大。然而，不同种系的密码子的应用统计学是可获得的，并可提示在概率的范围中最可能的逆翻译。

BLAST和FASTA动态翻译查询及数据库序列，因此在进行序列比较之前不用担心数据库的翻译。然而，如果需要算出一个6-框翻译的简单DNA序列，或者将一蛋白质逆翻译为一部分可能的DNA序列，又不想自己写脚本语言，EBI的Protein Machine Server（*http://www.ehi.ac.uk/translate/*）可为你做这部分工作。

成对序列比较

蛋白质和DNA序列的比较是生物信息学的基础之一。我们完成快速自动化比较的能力使下列任何事情都变得容易，包括一个新的序列的功能分配、模式蛋白质结构的预测与设计以及基因表达实验的设计与分析。随着生物学序列数据的累积，自然界是保守的这一点显得很明显。不会为每一个新的种系创造一个新的生物化学性质，新的功能特性也不会由全新基因的突然的出现而创造。事实上，不断的修改可导致遗传多样性及新功能的产生。将这个前提牢记在心，序列之间相似性的检测允许你将一个序列的信息合理地转移到另一个相似的序列，尽管它可能不总是全部的，但是一般来说，具有一定的可信度。

在进行核酸或蛋白质序列的可比较的陈述之前，序列比对结果是必需的。选择一个最佳的序列比对的基本概念很简单。两个序列可以任意的方式相匹配，将匹配的质量进行记分。然后，一个序列可以与另外一个进行匹配，并再次对匹配进行记分，直到最佳记分的比对出现。

在原则上听起来简单的东西在实际上并不简单。通过肉眼选择一个好的比对是可能的，但是人的生命太短暂了，一生恐怕只能完成一次或两次比对。很显然，从上千个选择中查找出最佳比对的自动化方法是正确的途径，但是为了使此方法稳定并具有生物学意义，需要回答几个问题。比对是如何进行记分的呢？记分方法可以像+1为匹配，−1为不匹配那样简单，但是什么是数据的最佳记分方法？序列中的空位是否应该开放使其他地方的匹配变得容易？如果空位是允许的，它们将如何记分？如果给定正确的记分参数，对于查找两个序列最佳的比对，最好的算法是什么？当一个比对已经完成时，它是否必须是有意义的？对于两个随机序列，能够产生相似质量的比对吗？本节下面，我们将详细讨论这些问题。

图 7-8 举例说明了三种类型的比对。这是使用一个叫做 ALIGN 的程序产生的三个成对序列的比对。在每个比对中，列出进行比较的序列，一个在另一个上面，可以排列出要匹配的残基。在正在匹配的残基之间，相同的匹配用冒号（:）表示，而相似的匹配用句点（.）表示。有关比对的信息在页面的上面显示，包括相似性的百分比及分值（相同匹配的数目用比对的长度相除）。最后，对由空位占据的序列的每一个位置，在一个序列中与另一个序列相关的空位用 –表示。

第一个比对是一个高分值的比对：它显示了两个密切相关蛋白质的比较（两个血色素分子，一个来自于海八目鳗，另一个来自于八目鳗类鱼）。和第2个比对进行比较，第二个是两个关系较远的蛋白质（也是两个血色素分子，用大米取代了八目鳗）。粗略的检查显示：低分值的比对和高分值的比对相比，共有的相同残基更少。而且，还有几个相似的或者是保守性的改变 ——其改变为一个氨基酸由另一个化学结构相似的氨基酸残基代替。第三种比对是随机比对，将两个不相关的序列进行比较（八目鳗血色素与人类的视黄醇结合蛋白）。值得注意的是，除了两个序列中很少的相同性及保守的突变之外，为进行比对两个序列中大的空位已经被打开。基因家族不可能以这个方式进化，如果给定的序列之间缺乏相似性，那么可以推论这些蛋白质是不相关的。

```
ALIGN calculates a global alignment of two sequences
 version 2.0uPlease cite: Myers and Miller, CABIOS (1989) 4:11-17
GENPEPT:AF248645_1                                     150 aa vs.
GENPEPT:AF156936_1                                     149 aa
scoring matrix: BLOSUM50, gap penalties: -12/-2
44.1% identity;         Global alignment score: 428

                        10        20        30        40        50        60
GENPEPT:AF24   MPIVDTGSVAPLSAAEKTKIRSAWAPVYSNYETSGVDILVKFFTSTPAAQEFFPKFKGLT
               ::::.: :  .  :: ..:  :: .:  .:.:.: .:. .:..:. ..:.::. :::::..
GENPEPT:AF15   MPITDQGPLPTLSEGDKKAIRESWPQIYQNFEQTGLVVLLEFLQKNPGAQQSFPKFSA--
                        10        20        30        40        50

                        70        80        9C       100       110       120
GENPEPT:AF24   TADQLKKSADVRWHAERIINAVNDAVVSMDDTEKMSMKLRDLSGKHAKSFQVDPQYFKVL
               :  .:... .:.:.: ::::::: ..  ::     :.. :..::.::.. :::::. :: :
GENPEPT:AF15   TKCNLEQDNEVKWQASRIINAVNHTIGLMDKEAAMKQYLKELSAKHSSEFQVDPKLFKEL
               60        70        80        90       100       110

                       130       140       150
GENPEPT:AF24   AAVIADTVAAGDAGFEKLMSMICILLRSAY--
               .:....:.  : :..:::.:.:: ::::.:
GENPEPT:AF15   SAIFVSTIR-GKAAYEKLFSIICTLLRSSYDE
              120       130       140
```

```
ALIGN calculates a global alignment of two sequences
 version 2.0uPlease cite: Myers and Miller, CABIOS (1989) 4:11-17
GENPEPT:U76030_1                                       166 aa vs.
GENPEPT:AF248645_1                                     150 aa
scoring matrix: BLOSUM50, gap penalties: -12/-2
20.6% identity;         Global alignment score: 94

                        10        20        30        40        50
GENPEPT:U760   MALVEDNNAVAVSFSEEQEALVLKSWAILKKDSANIALRFFLKIFEVAPSASQMF-SF--
               : .: :...::  .:  ... . ..:: . ..  . .. ...:.:  .:.:...: .:
GENPEPT:AF24   MPIV-DTGSVA-PLSAAEKTKIRSAWAPVYSNYETSGVDILVKFFTSTPAAQEFFPKFKG
                         10        20        30        40        50

                60        70        80        90       100       110
GENPEPT:U760   LRNSDVPLEKNPKLKTHAMSVFVMTCEAAAQLRKAGKVTVRDTTLKRLGATHLK-YGVGD
               : ..:  :.:.  .. ::  ..  . .:....  . :....    :. :.. : :  . :
GENPEPT:AF24   LTTAD-QLKKSADVRWHAERIINAVNDAVVSMDDTEKMSMK---LRDLSGKHAKSFQVDP
               60         70        80        90           100       110

                120       130       140       150       160
GENPEPT:U760   AHFEVVKFALLDTIKEEVPADMWSPAMKSAWSEAYDHLVAAIKQEMKPAE
                .:.:.  .. ::.              .: . ....:.. :   .. :
GENPEPT:AF24   QYFKVLAAVIADTV--------------AAGDAGFEKLMSMICILLRSAY
                  120                      130       140       150
```

```
ALIGN calculates a global alignment of two sequences
 version 2.0uPlease cite: Myers and Miller, CABIOS (1989) 4:11-17
GENPEPT:AF248645_1                                     150 aa vs.
GENPEPT:U13831_1                                       134 aa
scoring matrix: BLOSUM50, gap penalties: -12/-2
15.2% identity;         Global alignment score: -47

                        10        20        30        40        50        60
GENPEPT:AF24   MPIVDTGSVAPLSAAEKTKIRSAWAPVYSNYETSGVDILVKFFTSTPAAQEFFPKFKGLT
               :   : ...  . . :  ....   .  .. :  .  .    .:.: . ..    .:: :
GENPEPT:U138   MT-RDQNGTWEMESNE--NFEGYMKALDIDFATPKIAV---RLTQTKVIDQDGDNFKTKT
                         10          20        30           40        50

                                  70        80        90       100
GENPEPT:AF24   TA---------------DQLKKSADVRWHAERIINAVNDAVVSMDDTEKMSMKLRDLSGK
               :.               :.  :: : : :.. ...  .:..: ..  :: .   ..
GENPEPT:U138   TSTFRNYDVDFTVGVEFDEYTKSLDNR-HVKALVTWEGDVLVCVQKGEKENRGWKQW---
                   60        70        80         90       100       110

                 110       120       130       140       150
GENPEPT:AF24   HAKSFQVDPQYFKVLAAVIADTVAAGDAGFEKLMSMICILLRSAY
                   .. :  :...        . ::       ..: . .
GENPEPT:U138   ----IEGDKLYLEL---------TCGD--------QVCRQVFKKK
                         120                       130
```

图 7-8　三种比对：高分值、低分值但有生物学意义以及随机比对

在描述序列的比较时，通常使用几个不同的术语。序列相同、序列相似性及序列同源性是最重要的术语，每一个都有着轻微的不同，虽然它们经常可以相互替换地使用。

序列相同（ sequence identity）指的是两个比对的序列在相同的位置具有完全相同的核酸或者是氨基酸。序列相似性（ sequence similarity）只有当根据发生的概率，对可能的代替进行记分时才是有意义的。在蛋白质序列中，具有相似化学性质的氨基酸与不相似的氨基酸相比，更容易相互替换。这些倾向用于序列比对的记分的记分矩阵来表示。在两个氨基酸中，如果一个氨基酸被另一个替换，而从一个记分的矩阵中具有一正值的优势记分，就认为两个氨基酸具有相似性（在下一节描述）。

序列同源性（ sequence homology）是在描述序列之间进化上的相关性时更通用的术语。在比较两个序列时，通常是说序列同源性的百分数，尽管这个百分数是包括相同与相似的位点的混合物。最后，序列的同源性是序列之间进化上的同源性。如果两序列都来源于同一个共同的祖先序列，那么这两个序列被称为具有同源性。术语相似性和同源性在描述序列时经常相互替换，但是，严格地讲，它们的含义是不同的。相似性指的是两个序列中存在相同和相似的位点，而同源性强烈提示两个序列具有相同的祖先。

记分矩阵

在评估一个序列比对时，真正想了解的是一个给定的比对是随机的还有意义的。如果比对是有意义的，那么想测量它有意义的程度。可以通过建立一个记分矩阵来做这些工作。

记分矩阵(scoring matrix) 是描述残基（氨基酸或碱基）在比对中出现的概率值的表。在记分矩阵中的值是两种概率比值的对数，一个是在序列比对中氨基酸随机发生的概率。这个值只是指出每个氨基酸出现的独立几率的概率。另一个是在序列比对中，一对残基的出现有意义的概率。这些概率来源于已知有效的真实的序列比对的样本。

为了对比对进行记分，比对程序应该了解一个给定的氨基酸对是随机发生的，还是进化事件的结果。如果有意义的概率大的话，有意义的概率与随机发生概率比值的对数是正值，如果比对有意义的概率越大，其结果为负，因为得分是概率比值的对数，它可以为整个序列打分，比分越正，比对越有意义。

图 7-9 显示了 BLOSUM45矩阵的例子，BLOSUMA45 是一个非常流行的氨基酸的代替矩阵。

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	5	-2	-1	-2	-1	-1	-1	0	-2	-1	-1	-1	-1	-2	-1	1	0	-2	-2	0
R	-2	7	0	-1	-3	1	0	-2	0	-3	-2	3	-1	-2	-2	-1	-1	-2	-1	-2
N	-1	0	6	2	-2	0	0	0	1	-2	-3	0	-2	-2	-2	1	0	-4	-2	-3
D	-2	-1	2	7	-3	0	2	-1	0	-4	-3	0	-3	-4	-1	0	-1	-4	-2	-3
C	-1	-3	-2	-3	12	-3	-3	-3	-3	-3	-2	-3	-2	-2	-4	-1	-1	-5	-3	-1
Q	-1	1	0	0	-3	6	2	-2	1	-2	-2	1	0	-4	-1	0	-1	-2	-1	-3
E	-1	0	0	2	-3	2	6	-2	0	-3	-2	1	-2	-3	0	0	-1	-3	-2	-3
G	0	-2	0	-1	-3	-2	-2	7	-2	-4	-3	-2	-2	-3	-2	0	-2	-2	-3	-3
H	-2	0	1	0	-3	1	0	-2	10	-3	-2	-1	0	-2	-2	-1	-2	-3	2	-3
I	-1	-3	-2	-4	-3	-2	-3	-4	-3	5	2	-3	2	0	-2	-2	-1	-2	0	3
L	-1	-2	-3	-3	-2	-2	-2	-3	-2	2	5	-3	2	1	-3	-3	-1	-2	0	1
K	-1	3	0	0	-3	1	1	-2	-1	-3	-3	5	-1	-3	-1	-1	-1	-2	-1	-2
M	-1	-1	-2	-3	-2	0	-2	-2	0	2	2	-1	6	0	-2	-2	-1	-2	0	1
F	-2	-2	-2	-4	-2	-4	-3	-3	-2	0	1	-3	0	8	-3	-2	-1	1	3	0
P	-1	-2	-2	-1	-4	-1	0	-2	-2	-2	-3	-1	-2	-3	9	-1	-1	-3	-3	-3
S	1	-1	1	0	-1	0	0	0	-1	-2	-3	-1	-2	-2	-1	4	2	-4	-2	-1
T	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-1	-1	2	5	-3	-1	0
W	-2	-2	-4	-4	-5	-2	-3	-2	-3	-2	-2	-2	-2	1	-3	-4	-3	15	3	-3
Y	-2	-1	-2	-2	-3	-1	-2	-3	2	0	0	-1	0	3	-3	-2	-1	3	8	-1
V	0	-2	-3	-3	-1	-3	-3	-3	-3	3	1	-2	1	0	-3	-1	0	-3	-1	5

图 7-9　BLOSUM45矩阵，流行的氨基酸代替矩阵

氨基酸的代替矩阵是非常复杂的，因为它们反映了氨基酸的化学性质及出现的几率。比如在 BLOSUM 矩阵中，谷氨酸（E）如果用天冬氨酸（D）和谷氨酰胺（Q）所代替，其分值为正。这两个代替物在化学结构上是保守的。天冬氨酸具有一个与谷氨酸化学性质相似的侧链，虽然有一个甲基团短一些。另一方面，谷氨酰胺在大小及化学性质上与谷氨酸相似，但是它是中性的，而谷氨酸是带负电荷的。谷氨酸由异亮氨酸(I）和亮氨酸（L）残基代替，其分值为负。这些残基具有中性的、非极性的侧链；并且在化学性质上与谷氨酸不同。矩阵对角线的分值反映了每个氨基酸出现的几率。例如，如果其正值为 15，那么说明不可能是一个稀有的色氨酸（W）与另一个色氨酸的比对一致，然而最普通的丝氨酸（S）如果只有一个 4 分的正值，说明和另一个丝氨酸相匹配。记住这些值是对数是非常重要的，这说明两个丝氨酸的匹配不仅仅是一致而已。

DNA 或者 RNA 序列的碱基的代替矩阵非常简单。默认情况下，序列比对程序 BLAST使用为匹配指定一个标准的奖分、为不匹配指定标准罚分的策略，而不关心碱基的所有几率。在大多数的情况下，有理由推测 A:T 及 G:C 以大略相同的比例分布。

常用的代替矩阵包括 BLOSUM和 PAM矩阵。使用 BLAST 时，需要选择一个记分矩阵。大多数自动服务器会选择一个默认矩阵（通常像BLOSUM62)，如果想做一个快速的序列搜索，接受默认值是非常好的。

BLOSUM 矩阵来源于 Blocks数据库，一个没有空位的家族相关蛋白质的序列区域的比对集。聚类方法是将每一块中的序列分类为密切相关的组，在一个家族之内的序列之间代替的几率来源于有意义代替的概率。与 BLOSUM 相关的值（如 62）代表聚类步骤的分界值。值 62 表示如果序列有多于 62% 相同的话，序列将被放入同一个类。允许多样性序列包括进入每一个类的数目越多，分界值越低，代表进化时间段越长，因此，低的分界值矩阵对于寻找较远的亲缘关系是合适的。 BLOSUM62 对于没有空位的序列比对是标准的矩阵，而当产生具有空位的序列比对时，BLOSUN50 使用得更普遍。

PAM（point accepted mutation，点接受突变）矩阵根据相近关系序列比对的进化模型来测量。一个 PAM 单位等于在所有氨基酸位置中的平均变化是 1%。最通用的 PAM 矩阵是 PAM250 。然而使用 PAM 及 BLOSUM矩阵进行比较，结果提示对检测有生物学意义的相似性方面，BLOSUM 矩阵更好一些。

空位罚分

DNA序列改变不仅可以由点突变造成，也可以由残基的插入和删除引起。因此将空位引入一个或两个要排序的序列以产生序列之间有意义的比对是必需的。大多数算法使用空位罚分来代表在比对中加入一个空位的有效性。

空位的添加必须是理由充足的，而根据整个的比对分值，只有在真正需要时，空位才会被打开，并不是在整个序列中都被打开。大多数序列比对模型使用仿射(affine）空位罚分。其中打开序列中的一个空位的花费不同于扩展在开始时已存在的空位的花费。这两种罚分包括空位打开罚分以及空位扩展罚分，空位罚分的开放比相关的扩展罚分花费要多得多。这种倾向反映了同时发生几个残基的插入与缺失的倾向。

空位罚分与序列比对的记分矩阵密切结合：对于一个记分矩阵来说，最佳的空位开放与扩展罚分对不需要与其他的空位一起进行。一般空位 -BLAST 与 BLOSUM62 矩阵结合时空位开放记分为 −11 ,空位扩展为 −1，而 BLOSUM50使用 −12/ −1 罚分。

动态规划

动态规划方法是一个一般级别的计算法，经常用于序列比对及其他的计算问题。它们首先作为一种优化技术在 1950 年由 Princeton 大学的 Richard Bellman 描述。动态规划好像是由 Saul Needleman 及 Christian Wunsch 首先引入（注 2）生物学序列比较的。很明显，他们不清楚他们的方法与 Bellman 的方法之间的相似性。

与我们上面提及的一样，动态规划算法解决了优化问题，在许多可能的解决方案中只有一个（或一小部分）最好的方法。动态规划算法通过先将原始的问题分成小的下一级问题，然后再解决的方法寻找最佳的解决途径。这些大问题的部分具有次序的依赖性，也就是说，第四个部分可以只使用第三个部分的答案来解决，第三个部分可以只利用第二个部分的答案来解决，依次类推。动态规划通过下列方法运行，首先解决这些下一级问题，在一个表中存储装有分值的中间解决方案，最终选择产生最高分值的结果的顺序。动态规划算法的目标是使比对的总分值达到最大。为了做到这一点，高分值残基对的数目应该是最大的，空位及低分值对的数目应该最小化。

在序列比较中，全部的问题是寻找两个序列间最好的比对。这个问题被分为若干个比对的小问题，即将一个序列的每一个残基与其他序列的每一个残基进行比对；对于每个残基之间是否应该比对、序列的一个空位是否应该在第一个序列中引入，或者空位是否在第二个序列中引入，解决方法即可用来决定。每一个高分值的选择规则控制其他两个低分值的概率，以至于如果有关累积分数的信息在每一步中存储，每一个可能的比对不需要评估。

算法使用记分的（$m \times n$）矩阵（如图 7-10 所示），其中，m 和 n 是要比对的序列的长度，以一个空位和自身的比对开始（初始给定分数为零），算法一次填进矩阵的一列，在矩阵中的每一个位置，算法根据每三个选择中的结果分值进行计算，选择产生最高分的选择，然后再在当前位置至前面的通常达到高分的位置存储一个指针，当矩阵中的每一个位置都被填满之后，要完成一个追踪步骤，沿着指针的最高分值路径返回到比对的开头。

注 2：　也许应该像数学家们说的那样，是“重新发现”。因为计算生物学结合了许多领域中的研究，这种相互独立的发现经常会发生，只是在后来才被人们注意到。

	-	g	c	t	g	g	a	a	g	g	c	a	t
-	0	0	0	0	0	0	0	0	0	0	0	0	0
g	0	5	0	0	5	5	0	0	5	5	0	0	0
c	0	0	10	3	0	1	1	0	0	1	10	3	0
a	0	0	3	6	0	0	6	6	-g	0	3	15	8
g	0	5	0	0	11	5	0	2	11	5	0	8	11
a	0	0	1	0	4	7	10	5	4	7	1	5	4
g	0	5	0	0	5	9	3	6	10	9	3	0	1
c	0	0	10	3	0	2	5	0	3	6	14	7	0
a	0	0	3	6	0	0	7	10	3	0	7	19	12
c	0	0	5	0	2	0	0	3	6	0	5	12	15
t	0	0	0	10	3	0	0	0	0	2	0	5	17

g a a g - g c a

g c a g a g c a

图 7-10 比较两个序列的记分矩阵 其中高分匹配突出显示

全长比对

可能会遇到的一个比对案例就是两个序列的全长比对。整个序列比对的算法称为 Needleman-Wunsch 算法。在这个情况下，最佳的比对是从下一级序列的高分值比对中建立，通过从顶端左侧到底部右侧的矩阵一步一步地进行。只有最好的分值可以通过矩阵跟踪时，才会产生最佳比对。

使用 ALIGN 产生一个全长序列比对

既然已知两个序列全长比对的理论，让我们检测一下实现这个算法的程序。 ALIGN 是计算全长比对的简单工具。它是在本章后面描述的 FASTA 软件的一部分，在 FASTA 中的程序从 Unix 命令行很容易运行，但是如果想通过点击界面访问它们，可通过已整合进 SDSC 生物学工作台（Workbench）的以 Web 为基础的序列分析软件来访问。FASTA 程序在 Linux 下很容易编译；然而，一旦编译完成，需要将它们链接进入 */usr/local/bin* 目录或其他有意义的方便的位置。

为运行 ALIGN 或者任何其他的 FASTA 程序，需要 FASTA 格式的序列数据。这是

最常用的序列格式之一，而且可能是最简单的。为了使用 ALIGN，正在比对的每个序列都应该是一个单独的文件。

FASTA 格式（注 3）的序列如下：

```
>2HHB:A HEMOGLOBIN (DEOXY) - CHAIN A
VLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHFDLSHGSAQVKGHGK
KVADALTNAVAHVDDMPNALSALSDLHAHKLRVDPVNFKLLSHCLLVTLAAHLPAEFTPA
VHASLDKFLASVSTVLTSKYR
```

FASTA 格式非常灵活，它是序列分析软件最常用的格式之一。 FASTA文件包含一个或更多个 FASTA 格式的记录，由空行分开。每一个记录包含一个人类可以阅读的评论，随后是一个核苷酸或蛋白质序列。评论显示在记录的第一行；必须以＞号开始，随后是序列的一个或者是多个标识符。评论可能包括由序列代表的分子信息，如蛋白质或基因的名字及其结构来源。在以前的例子中，标识符是一个 PDB 代码（2HHB），随后是有关序列的描述（一个脱氧血色素蛋白的 A链）。记录的剩余部分包含序列自身，被换行符分成单独的行。每行通常有 60 个字符长，但是其格式不指定一行的长度。提取 FASTA 数据作为输入的程序（比如 ALIGN），通常允许 FASTA 的自由形式的性质。检查程序的文档以确定数据是否已恰当格式是一个很好的习惯。

要使用 ALIGN，可在命令提示符下直接输入*align*。然后将提示建立序列文件名。结果输送至标准输出。尽管使用更容易解析的比对格式是一种趋势，由 FASTA 产生的成对比对 ASCII 格式仍然很常用：

```
Output scoring matrix: BLOSUM50, gap penalties: -12/-2 43.2% identity;
Global alignment score: 374

                 10        20        30        40                50
2HHB_A V-LSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHF-DLS---HGSA
       : :.: .:. : : ::::  .. : :.::: :... .: :. .:  : :::      :.
2HHB:B VHLTPEEKSAVTALWGKV-NVDEVGGEALGRLLVVYPWTQRFFESFGDLSTPDAVMGNP
                10        20        30        40        50
              60        70        80        90       100       110
2HHB_A QVKGHGKKVADALTNAVAHVDDMPNALSALSDLHAHKLRVDPVNFKLLSHCLLVTLAAHL
       .::.::::  :.....::.:..  .....::.::  ::.::: ::.::.. :. .:: :.
2HHB:B KVKAHGKKVLGAFSDGLAHLDNLKGTFATLSELHCDKLHVDPENFRLLGNVLVCVLAHHF
       60        70        80        90       100       110
           120       130       140
2HHB_A PAEFTPAVHASLDKFLASVSTVLTSKYR
         :::: :.:. .: .:.:...:. ::.
```

注 3： 也称为 Pearson格式，即以 FASTA 软件的作者 William Pearson命名。

```
2HHB:B GKEFTPPVQAAYQKVVAGVANALAHKYH
         120        130        140
```

FASTA 发布版包含使用局部成对比对程序 LALIGN的 HTML表单和CGI脚本样例。如果要求基于 Web的界面的话，脚本语言可以在修改后应用于 ALIGN 程序。

局部比对

最通用的序列比对工具依赖于一个称为局部比对（local alignment）的策略。以前讨论的全长比对策略假定要比对的两个序列是已知的，而且对全长进行比对。然而，在进行序列比对时经常遇到的情况是，或者是用一个序列在序列数据库中寻找未知序列，或者是在一个很长的 DNA序列，如基因组的一部分中，寻找与查询序列相匹配的部分。在确实具有进化相关性，但是彼此多样性比较明显的蛋白质或基因序列中，短的同源性序列片段可能是保留下来的序列同源性的所有证据。

执行两个序列的局部比对的动态规划算法称为 Smith-Waterman 算法。以它的创造者 Temple Smith 博士及 Michael Waterman博士命名，除了通过矩阵跟踪时允许一个附加的选择之外，这个算法与 Needleman-Wunsch 算法相似。局部比对不需要对要进行比对的两个序列从头到尾进行。如果在序列中到某点的累积积分达到负值的话，应该抛弃这个比对，并开始一个新的比对。这个比对也可以在矩阵的任何位置终止。

局部比对的工具

Smith-Waterman 算法最常用于数据库搜索的实现是SSEARCH程序，它是后面描述的 FASTA 发布版的一部分。LALIGN，也是 FASTA 软件包的一部分，是 Smith-Waterman 算法用于两个序列比对的实现。

在生物学数据库进行序列查询

序列比对的一个常见应用是在数据库中搜索与查询序列相似的序列。在这些搜索中，一个具有成百上千个残基的序列的比对，至少要与数据库中上万个大小相似的序列进行匹配。如果没有具有特殊目的的硬件可用，使用基于动态规划的方法不是很方便。而对于常规的搜索来说，使用特定的试探法的数据库搜索方法。试探法的方法利用有关序列和比对统计的知识使得大范围的搜索变得有效及实用。虽然它们不能

保证最佳选择，但是它们使大的序列数据库的敏感搜索成为可能。在本节中，我们将描述 BLAST及 FASTA 这两个常用的数据库搜索程序。

使用 BLAST 进行以局部比对为基础的搜索

迄今为止，最常用的搜索序列数据库的程序是 BLAST（Basic Local Alignment Search Tool ,基本局部比对搜索工具)。BLAST是大多数在线序列搜索服务器的核心算法(注 4)。它可执行序列的成对比较，寻找具有局部相似的区域，而不是整个序列之间最佳的全长比对。

BLAST可以在几分钟之内进行上百个甚至上千个序列比较。在几个小时之内，一个查询序列可以与整个数据库进行比较，以找到所有相似的序列。 BLAST由于这个功能的实现，使它变得非常流行，以至于在计算生物学社区它已成为一个动词，如“ I BLASTed this sequence against GenBank and came up with three matches.（使用 BLAST 在 GenBank 中搜索出三个匹配的序列)”。

BLAST算法

使用标准的 Smith-Waterman 算法进行局部序列比对搜索是一个相当缓慢的过程。BLAST算法可以加速局部的序列比对，它具有 3 个基本的步骤。第一，在比对查询序列时，创建一个包括所有分值超过阈值的短序列（在 BLAST 术语中称为 Words）的列表。第二，在序列数据库搜索这些短序列。因为这些短序列非常短（蛋白质为 3 个残基，核酸为 11 个残基），为了提高速度，搜索所有短序列的预计算表，以及它们在序列中的位置是可能的。然后，这些匹配的短序列再延伸至查询序列与数据库序列之间没有空位的局部序列。这种延伸一直延续到比对分值降低至一个阈值。序列中最高分的比对，或者称为 MSP（maximal-scoring segment pair）,与可能的位置结合成为局部比对。最初， BLAST 只搜索没有空位的序列的比对。然而，新附加到 BLAST 软件包的内容已经引入了对有空位的序列比对的搜索。

NCBI BLAST和 WU-BLAST

BLAST算法有两个实现：NCBI BLAST和 WU-BLAST 。两个实现都可用于 Web服

注 4: 生物信息学的常用工具已经存在很长时间了，这里给你一个感性认识：最早的 BLAST 论文是由 Altschul 等在 1990 年 10 月的《Journal of Molecular Biology》上发表的。

务及可下载的软件包。NCBI BLAST 可从 NCBI 处获得，而 WU-BLAST 是来源于 NCBI BLAST 1.4 的一个替代版本，由华盛顿大学的 Warren Gish博士及其同事开发和维护。

在两个实现中，NCBI BLAST 更加常用。此程序的最新版本着眼于开发多序列的功能评估（见第八章）方法。另一方面，WU-BLAST 已开发了处理空位以及许多特征（比如重复滤过）的不同的系统，对搜索基因组序列很有用。后来，TIGR，斯坦福大学酵母基因组服务器，伯克利大学的果蝇基因组计划等通过 Web 使用 WU-BLAST 2.0 作为搜索基因组序列数据的序列比较工具。在本书写作期间，WU-BLAST 2.0 这个最新的软件版本已经注册了版权。NCBI BLAST 及 WU-BLAST 以前版本即 1.4 版本，都是公共软件，所有研究人员都可以免费获得。由于 NCBI BLAST的应用具有普遍性，在下面的部分我们将主要讨论它，但是 WU-BLAST也是可选择的。如想浏览 BLAST的更多信息，请访问 NCBI 的站点：*http://ncbi.nlm.nih.gov*或者 WU-BLAST 的站点 *http://blast.wustl.edu*。

各种 BLAST 程序可做什么？

经常使用 BLAST的用户可以在他们自己的计算机上下载并安装BLAST。BLAST在 Linux 系统上很容易安装。直接创建一个新的目录（如，*/usr/local/blast*），拷贝压缩包并解压缩。这里是 BLAST发布包中的 4 个主要的可执行程序：

[blastall]

应用 5个BLAST 程序之一：*blastp*、*blastn*、*blastx*、*tblastn*或 *tblastx*完成BLAST 搜索

[blastpgp]

在 PSI-BLAST 或 PHI-BLAST 模式下完成搜索

[bl2seq]

完成两个序列的局部比对

[formatdb]

将一个 FASTA 格式的平面文件序列数据库转换为一个 BLAST数据库

blastall 包含没有空位及有空位的序列 BLAST 搜索的所有主要的选项。用命令 *blastall-* 可以显示命令行参数的全部列表：

[-p]

程序名。它的选项包括：

blastp

在蛋白质序列（PS）数据库中进行 PS 查询

blastn

在核酸序列（NS）数据库中进行 NS 查询

blastx

在 PS 数据库中将 NS 查询翻译成所有 6 个阅读框

tblastn

在 NS 数据库中进行 PS 查询，动态翻译成所有 6 个阅读框

tblastx

已翻译的 NS 在已翻译的 NS 数据库中进行查询，进行透彻的计算

[-d]

数据库名。每个索引数据库包含几个文件；名字是那些文件名的普通部分

[-i]

Query 序列文件名。如果不给定，则默认为标准输入

[-e]

期望值被切断（参见后面“评估 BLAST 结果”一节）

[-m]

比对查看。它的选择包括：

0 成对

1 Master-Slave，显示相同

2 Master-Slave，不相同

3 Flat master-Slave，显示相同

4 Flat master-Slave，不相同

5 Master-Slave，不相同，硬终止

6 Flat master-Slave，不相同，硬终止

[-o]

输出文件名。如果不特别指出，默认为标准输出

[-G]

空位开放罚分。默认为 11（适于 BLOSUM 63矩阵）

[-E]

空位扩展罚分。默认为 1（适于 BLOSUM 63 矩阵）

[-q]

核苷酸不匹配罚分。只适用于 *blastn*。默认为 -3

[-r]

核苷酸匹配奖分。只适用于 *blastn*。默认为 1

[-b]

显示比对的数目。

[-g]

完成有空位的序列的比对。T/F。使用 *tblastx* 时无此选项

[-M]

记分矩阵。默认为 BLOSUM 62

[-T]

产生 HTML 输出。T/F。默认为 F

这些是可能经常使用的命令行选项，但还有大量的其他选择可以利用。在熟悉 BLAST 后，大家可能就想去学习使用它们。

blastpgp 允许使用两个新的 BLAST 模式：PHI-BLAST（Pattern Hit Initiated BLAST）及 PSI-BLAST（Position Specific Interative BLAST）。PHI-BLAST 使用蛋白质基序，比如在 PROSITE及其他基序数据库中发现的，以提高找到有生物学意义匹配的概率。PSI-BLAST使用迭代序列比对过程开发与位置有关的记分矩阵，用来提高检测弱模式匹配的能力。在第八章分析多序列时，我们会对这两种方法做进一步的讨论。

bl2seq 允许使用 *blastp* 或 *blastn* 程序对两个已知序列进行比较。对于 *bl2seq* 来说，大多数命令行选项都和 *blastall* 的选项相似。

用 formatdb 建立一个局部数据库

用法：`formatdb -i araseed.nt -p F -o T`

尽管在许多 Web服务器上都可获得BLAST ,但是在本地安装及使用 BLAST的好处之一，是具有创建自己的序列数据库的能力。比如，你可能有一部分序列还未发表或者没有公开发表。它们不在 GenBank数据库中，因此如果不能在自己的计算机上运行 BLAST，你怎样去搜索它们？

程序 *formatdb*接受 FASTA形式或 NCBI的 ASN.1形式的序列数据库输入（参见第十二章）。不论其输入是蛋白质还是核酸序列，不论是否为数据库创建索引，在程序命令行指定输入文件名都是必需的。还有其他命令行选项可用，这些选项可以在没有指定命令行选项时，通过运行*formatdb* 查看。

创建的文件为：

```
araseed.nt.nhr
araseed.nt.nin
araseed.nt.nsd
araseed.nt.nsi
araseed.nt.nsq
```

评估 BLAST 结果

在序列数据库中进行一个 BLAST搜索，会产生几十个甚至上百个候选的比对。在这些比对中，怎样才能知道哪一个是真正有意义的同源性，哪一个只是与不相关序列最好的匹配？ BLAST 提供了 3个相关的信息以允许对结果进行解释：原始记分、位记分及 E- 值。

局部序列比对的原始记分（raw score ）是组成比对的 MSP 的分数总和。因为记分矩阵不同，不能总是通过原始记分进行直接的比较。位记分（bit score ）是原始记分已经进行从创建比对的记分矩阵的 log 为底到 log 以 2 为底的转换，这种重新调节允许对记分比对之间进行比较。

E- 值（ E-value）提供有关一个给定序列比对具有意义的概率信息，比对的 E- 值提示希望找到的得分大于或者等于在随机数据库中搜索时的比对分值的比对数目。因此一个大的 E-值（5或 10）提示的比对可能是由于偶然发生的、靶序列与数据库中

的不相关序列进行了比对。 E 值为 0.1 或 0.05 在序列数据库搜索中常用作分界（cutoff）。如果在数据库搜索中使用一个大的E值分界，则会有许多关系较远的匹配也会被找出，但它是一个具相对高的假阳性的比对结果，在这三个值当中， E- 值是最常在文献中被报道的值。

这里有一个界限，如果越过了这个界限，那么对比较的序列的相关性来说，序列相似性信息将变得没有价值。对一般长度的蛋白质序列来说，这个界限大约是低于 25% 的序列相似性，尽管研究工作还在继续推动这个界限。如果序列相似性低的蛋白质序列仍被认为具有序列相关性，那么结构分析技术应为这种关系提供证据。当序列的结构为未知时，具有低相似性的序列被分类为不相关的，但是其含义也许是序列之间进化的距离太远，以至于这种关系不能检测出来。

使用 FASTA 进行局部比对

局部序列比对的另一个试探法方法是 FASTA 算法。FASTA 算法比 BLAST 算法出现得还早，现仍由 Virginia大学的 William Pearson 博士积极地维护着。像 BLAST 一样，FASTA 可以作为 Web 服务，也可是一组可下载的程序。在本节中，我们将讨论当前 FASTA 算法的版本以及包括在 FASTA 发布包中的一些程序。

FASTA 算法

FASTA 首先搜索查询序列及序列数据库中均出现的短序列（称为 ktup，注 5）。然后，算法使用 BLOSUM50矩阵对 10个包含最相似的 ktup 没有空位的比对记分。检测这些没有空位序列的比对在它们的分数不低于一个阈值时，并入一个有空位的序列的比对的能力。对于那些分值在阈值以上的序列比对进行并入，对那段区域的最佳局部比对进行计算并报告比对分数（称为最佳分数）。

FASTA ktup 比 BLAST word 要短，对蛋白质来说通常为 1 或 2，对核酸来说为 4 或 6，低 ktup 值导致缓慢但是敏感的搜索，而高 ktup 值会产生快速的搜索，其错误阳性率低。

注 5: k 元组（k tuple）或 k 残基的有序序列的编写。

FASTA 程序

FASTA发布包包括与主要的 BLAST模式相似的搜索程序，也包括全长及局部成对的比对及其他有用的功能的程序，但是 PHI-BLAST及 PSI-BLAST 除外。在此处列出的 FASTA程序都很容易在 Linux系统中编译：

[fasta]

使用 FASTA 算法将一个蛋白质序列与蛋白质数据库（或将 DNA 序列与 DNA 数据库）进行比较。

[ssearch]

使用 Smith-Waterman算法将一个蛋白质序列与蛋白质数据库（或将DNA序列与 DNA 数据库）进行比较。

[fastx/fasty]

将一个 DNA 序列与一个蛋白质数据库进行对比，对 DNA 序列进行翻译。

[tfastx/tfasty]

对 DNA 序列数据库进行翻译，将一个蛋白质序列与一个 DNA 数据库进行比较。

[align]

在两个 DNA 序列或蛋白质序列之间计算全长比对。

[lalign]

在两个 DNA 序列或蛋白质序列之间计算局部比对。

在写本书时，这些程序除 ALIGN 和 LALIGN 之外，都包括在FASTA 3.3 软件包中，ALIGN及 LAIGN 只包括在 FASTA 2 中。

序列分析的多功能工具

多个研究团体及公司已经装配基于Web的到序列工具的集合的界面。其中最好的是完全整合工具、公共数据库、保存使用者数据记录的能力以及从一个用途到另一个用途的功能。如果要搜索一个或几个序列的匹配情况，而且想搜索标准的公共数据库，这些入口可帮助你节省许多时间，同时可以提供大多数功能特性并使使用商品

化的序列分析软件变得容易。在有些情况下，不得不为一个许可证或认购付费，以得到这些站点的全部功能特性；而有些站点是可免费访问的。

NCBI SEALS

NCBI SEALS 计划旨在为大量序列的系统分析开发一个以Perl为基础的命令行环境。SEALS 远远不是一个完全自动化的基因组分析工具，而且它也不准备这样做。SEALS 能提供的是强化 Unix 系统中的命令行环境。它由多种带有各种有用功能的脚本语言组成：转换文件格式、对 BLAST 结果及 FASTA 文件进行操作、数据库检索、将文件管道输送进 Netscape以及在不需要吸取资源的 GUI 时，使数据更容易读取的一系列其他性质。 SEALS 在 Unix 系统上运行，可能对那些 Unix 爱好者最有用。在写脚本语言来加工大量的序列之前，检查一下看你所想要的过程在 SEALS 中是否实现。

生物学工作台

SDSC（圣迭戈 San Diego 超级计算中心）通过生物学工作台（Biology Workbench）提供进入序列分析工具的入口。自从 1995年以来，这个资源一直以不同的形式为学术研究人员提供免费服务。用户在 SDSC 站点上可以得到登录账号和一个密码，而且工作会话及数据库在服务器上可以保存得相当安全。

生物学工作台提供大约 40个主要的序列数据库，以及超过 25个全部基因组进行以关键词和序列为基础的搜索。 BLAST 及 FASTA，与几个局部的或全长比对工具、DNA序列翻译的工具、蛋白质序列特征的分析、多序列比对以及系统进化树的绘制工具一起，在工作台中作为搜索与比对工具得以实现。工作台群还没有实现简图的工具，如 MEME、HMMer 或序列日志工具，尽管在序列搜索时可使用 PSI-BLAST。

尽管生物学工作台的界面有些麻烦，包括许多滚动窗口以及需要点击的按钮，但是它仍旧是应用最广泛、最方便、以 Web为基础的工具箱。它主要的好处之一，是软件可以接受和翻译许多种序列文件格式。工作台的用户从来不用担心文件类型的不兼容，可以从以关键词为基础的数据库搜索无缝转移至以序列为基础的搜索、多序列比对以及种系发育分析。

Double Twist

另一个进入序列分析的入口是 Double Twist 站点 *http://doubletwist.com*。这个站点允许提交一个序列，然后用 BLAST 在多个数据库中进行比较。为了新添加匹配的提交序列和通知用户，它们提供了代理来监视数据库。这些服务和进入 EcoCyc 数据库，以及在线生物学研究协议数据库，在本书写作时，可以免费注册。

第八章

多序列比对、进化树和简图

在第七章中，我们提出了利用序列比对来寻找和比较相关序列对的思想。然而令人感兴趣的生物学问题往往涉及两个以上序列的同时比较。例如，通过 BLAST 或 FASTA 搜索可能会得到大量符合检索条件的序列。怎样对得到的所有序列相互进行比较呢？或者说，怎样通过检查这些序列来理解它们之间的相互关系呢？

一个方法是在所有序列对间进行成对比对，并分别对这些成对比对进行研究。然而，一个更有效（也更容易理解）的方法是，同时比较所有序列，之后检验全体比对。这种方法称为多序列比对（multiple sequence alignment）。多序列比对可以通过对相关的基因或蛋白质群的研究来推理基因间的进化关系，并发现结构或功能相关的序列群中共有的模式。在本章中，我们将介绍一些建立和解释多序列比对的软件工具，并描述它们的一些应用程序，其中包括系统发育推论和基序的发现。系统发育推论和基序的发现是进化理论的基础，因此，在讨论进化领域的生物信息学之前，首先让我们花一点时间来回顾一下进化的历史和理论。

从形态到分子

为了详细讨论多序列比对，让我们再简单回顾一下进化。一直以来生物学的目的之一是创立生物的分类方法，也就是根据生物间的相互关系来组织物种的方法。早期的生物学家仅仅按照形态学（即生物的身体外观）来划分物种。后来，随着解剖方法的普及，生物学家开始根据解剖学对生物进行分类。

博物学家早已经发现一些生物的化石，它们不像当时的任何一种活着的生物，但他们认为这些已成为化石的生物曾经在地球上出现过。这些证据表明地球上的生物可能一直处于变化之中。这也表明物种之间的关系不是静止的，而极有可能是一种进化过程的结果。随着对地球地质物理过程的了解，人们对发生这种变化所需的时间也越来越清楚。目前科学家广泛接受的观点是：地球生命起源于大约35亿年前。因为根据目前发现和整理的化石纪录，推测最早出现的生物是35亿年前的一种类似于细菌的单细胞生物。

最终被大多数生物学家接受的进化理论是查理·达尔文的进化论。达尔文认为生物的每一代都会发生变异。有些变异有利于个体在环境中的生存，这些个体可以更成功地繁殖后代，从而使更多的后代具有这种有利性状。根据这个理论，地球上生命形式的多样性来源于变异，甚至可能是来源于一个共同的单细胞祖先的变异，物种经过变异来适应各种不同的生态环境，从而形成了生物的多样性。

分子进化把进化的概念扩展到了DNA和蛋白质序列层次。尽管DNA序列的复制是一个十分精确的过程，但是伴随着辐射损伤和其他基因组序列的突变或改变，小的复制错误在不断积累。进化压力对生物的选择不是根据形态特征进行的，而是发生在突变水平上。因此，健康生物中可以观察到的突变是那些不会引起生物死亡的突变。

因为基因序列间的这些改变是递增的，所有我们可以从许多不同生物中得到同源基因（具有共同进化起源并且功能相关的基因）并通过比对相同或者相似的的残基来比较它们。这种多序列的比较显示一个基因（或它的蛋白质产物）的哪些区域对突变是敏感的，哪些区域对氨基酸的替换是可以忍受的。因此我们可以对这些序列在进化中所发生的分子事件做出假设。许多生物信息学的方法，包括序列的成对比较和序列数据库搜索，都是基于同源基因具有相似的序列这一观测资料进行的。

在考虑序列的相似性时，我们心中会有另一个疑问：那就是直向同源（orthologs）和共生同源（paralogs）之间的不同。分子进化的化学过程不仅仅会引起物种的分化。进化变化也会发生在单个物种的基因组中。直向同源是与进化相关的、具有相同的功能、因物种生成而分离的一组基因。另一方面，共生同源具有共同的祖先，但是因基因复制而产生不再有相同功能的基因。换而言之，直向同源具有相同的功能，但是出现在不同的物种间。而共生同源存在于相同的基因组中，但具有不同的

功能。序列的数据库搜索可能会返回直向同源和共生同源结果。根据研究目的的不同，你可能不希望把它们所有的结果都放在一个组中。

多序列比对

多序列比对技术在蛋白质序列的研究中应用最广泛。理想的结果是它们表明比对中每一个序列编码的蛋白质具有进化和结构的相似性。我们知道在生物之间功能相近的蛋白质具有序列和结构的相似性。在进化过程中序列的改变快于结构的改变。在只从序列数据得到的多序列比对中，常常发现序列中相似的区域在结构上也具有相似性。

凭着对蛋白质生物化学性质的详细了解，我们可以手工建立多序列的比对。然而这是一个艰苦的过程。自动多序列比对的问题在于很难精确定义什么是最佳的多序列比对，也不可能为一个正确的多序列比对建立一套标准。理论上讲，在一组序列中有一个底层进化过程和一个进化上正确的比对。但是，在比对的某些部分可能序列间的差别太大，很难利用比对算法得到一个明确、单一的答案。这些同样的多变区在结构上往往也是不可比对的。从多序列比对得到的大多数观点来源于对同源区的分析，而不是试图比对多变区。

成对序列比对所用的动态规划运算法，理论上可以扩展到任意多个序列。然而随着序列数目的增加，运算所需的时间和内存会呈指数增加。两个序列的动态规划比对只需几秒钟。对 4 个相对较短的序列进行比对则需要几小时。因此，用这种方法比对多序列是不可行的。程序 MSA 是在一定程度上减少了多序列比对的动态规划问题的复杂性的算法的一个实现。在可以接受的时间内，它可以排列 7 个相对较短的蛋白质序列（200~300 氨基酸）。可是在大量序列的比较中 MSA 就很少起作用了。

多序列比对的渐进策略

多序列比对的一般方法是渐进地比对对序列。一般的渐进策略如下：选择和比对一对起始序列，然后接下来的每一个序列与前一个比对进行比对。与用于序列比对的 Needleman-Wunsch 和 Smith-Waterman 算法一样，渐进比对方法是一种试探法算法。更进一步说，是一种贪婪算法。贪婪算法（greedy algorithm）是把一个问题分

成许多部分，找到每一部分的最好解决方案，而不必去考虑问题的整体。具体到渐进比对，则是把整个问题（许多序列的比对）分解为一系列序列成对比对的过程。

因为是一种试探法算法，所以渐进比对不能保证会得到可能的最好比对。然而，实际上，这是一种有效的方法，可以得到有生物学意义的结果。渐进比对在以下方面有所不同：是把后来的序列与累积的比对进行比对还是创立一个亚族，是如何选择起始的一对序列进行比对；如何给单个比对和单个序列与原来比对的比对打分。

利用 ClustalW 进行多序列比对

ClustalW是一个常用的多序列渐进比对程序。 ClustalW中的试探法是建立在系统发育分析的基础之上的。首先，产生一个所有待比对序列的成对距离矩阵，并通过邻近加入算法产生一棵向导树。然后，每个关系最近的序列对（进化树最外面的分支）用动态规划法进行相互比对。然后，通过分析每个新比对来建立一个序列简图。最后，比对简图之间相互比对或和其他序列进行比对（取决于进化树的拓扑结构）直到建立完整的比对。

在一定的条件范围内，这个策略可以产生合理的比对。但并不是这么简单，对于相关性较差的序列，它可能是建立在错误的成对比对和错误的系统发育分析基础之上的。但是对有明显相关序列对的序列组，这个方法是可靠的。对于相关性很好的序列，无论使用得分矩阵还是罚分值（ penalty value），动态规划法所得到的成对序列比对都是十分准确的。对相近序列的系统发育分析相对明确一些。对相关性较远的序列利用多序列创建简图可以增加成对比对分析的准确性。

多序列比对分析的相关参数很多。当然，包括与成对比对步骤中相关的得分矩阵和空位罚分（ gap penalty），此外还有一些改变序列 －简图和简图 －简图比对中得分矩阵的权重参数。在ClustalW 中它们设置在Multiple Alignment或Profile Structure子菜单中，在 ClustalX 中它们则位于 Alignment下拉菜单中。

在多序列比对分析中的成对比对参数大家已经熟悉了，它的意义与成对比对分析中的相同。多序列比对参数包括多序列比对过程中的空位打开和扩展罚分（用于调整比列）以及在开始序列比对时用于起始发散序列的最大允许延迟，由序列长度决定。

蛋白质序列比对分析中，ClustalW 的一种试探法是，对于每一个比对，根据预期的

进化距离使用不同的得分矩阵。如果进化树中的两个序列是很近的邻居，可以通过一个优化的用于相近关系的得分矩阵进行比对。因此，在 Multiple Alignment Parameters 菜单中选出一系列矩阵时，这仅仅意味着：BLOSUM62 用于相近关系，BLOSUM45 则用于较远关系，而不是对所有的成对比对都使用相同的得分矩阵。

ClustalW使用的另一种试探法是蛋白质简图比对的可伸缩空位罚分。靠近保守的疏水性残基的空位会比靠近亲水性残基的空位罚分更多。太靠近另一个空位的空位会比单独的空位罚分更多。这些参数在 Protein Gap Parameters 菜单中设置。

虽然ClustalW通过Unix命令行运行，但它是菜单驱动的，并不依靠命令行选项。要启动程序，只需键入 *clustalw*，就会出现一个选择菜单：

```
**************************************************************
******** CLUSTAL W (1.8) Multiple Sequence Alignments  ********
**************************************************************
     1. Sequence Input From Disc
     2. Multiple Alignments
     3. Profile / Structure Alignments
     4. Phylogenetic trees

     S. Execute a system command
     H. HELP
     X. EXIT (leave program)
```

这个菜单及输入序列后出现的子菜单，将以一种相当直观的方式引导大家使用 ClustalW。比对将用纯文本格式给出。

尽管在 Linux 工作站上安装和使用 ClustalW都很简单，但是 ClustalX（基于 X-Window的ClustalW 用户图形界面）并不容易编译。然而， ClustalX运行于自己的 X-Window， 有自己的下拉菜单，允许以彩色标记的方式查看和绘制多序列比对的结果。它也允许向一个比对中逐个添加序列，允许以彩色的 PostScript 形式输出来自不同文件的比对中特殊的序列区间，以及其他一些方便的性质。要在 Linux 上安装 ClustalX，需要有：

ClustalX 二进制文件

NCBI 发布的软件工具包源程序

LessTif 库

首先要做的是安装 LessTif库。这个发布包非常易于使用。LessTif 库可以从 *http://www.lesstif.org*下载，也可以从 Linux 发布包中得到。NCBI 工具包可以从 *http://www.ncbi.nlm.nih.gov*得到，只要 LessTif库一安装到 */usr/X11R6/lib*，该工具包就可以完全进行编译。如果 NCBI 工具包安装产生一个 *libvibrant.a* 文件，*clustalx* 命令就可以正常工作了。

利用 Jalview 查看和编辑比对

用法：`Jalview` *`alignmentfile FORMAT`*

查看比对是很有用的，但是我们常常也需要对比对进行编辑。比对编辑是极少数在 Windows 平台上比在 Unix平台上实现得更好的生物信息学功能之一。但如果在工作站上安装了对Java语言的支持，可以使用Jalview程序编辑比对,该程序由 Michele Clamp编写，可从 EBI（*http://www.ebi.ac.uk/~michekle/jalview/content.html*）处得到（注 1）。

为了在Linux 工作站上使用 *Jalview*，必须下载完整版的 Unix。打开软件包,然后编辑 *Jalview*文件以反射本地环境。Jalview 是为 *Jalview* 设置环境并实际启动程序的脚本。例如，设置环境变量 CLASSPATH 来反射 JDK（Java开发包）中的类文件和 *Jalview* 类文件（*jalview.jar* 文件）的位置。设置环境变量 JAVA.EXE 指向 Java 可执行文件：

```
setenv CLASSPATH /usr/jdk118/lib/classes.zip:/usr/local/jalview/jalview.jar
setenv JAVA_EXE /usr/jdk118/bin/java
```

Jalview可以从 Clustal 中读取比对（*.aln*）文件和其他几种格式的比对文件。我们主要讨论 Jalview的比对编辑功能，但你也可以发掘其他一些功能。如果把它作为命令行选项，它也可用来搜索数据库。

注 1： 安装 Java支持软件包括 JDK（Java Developement Kit）和 JRE（Java Runtime Environment）。IBM 公司已经将二者移植到 Linux 上了。可以从 IBM 的网站 *http://www.ibm.com/java/jdk/index.html*获得。可以注册后下载，但这是免费的。软件包是易于安装的 RPM形式。虽然在生物信息学中不会遇到很多 Java程序，但是如果碰到了，我们就可以高枕无忧。

要运行 Jalview，应在工作目录中建立一个 Jalview 脚本的链接。

Jalview识别的比对文件格式通常是 MSF、CLUSTAL 和 FASTA 。在命令中，它们都要用大写字母表示。

Jalview 窗口是激活区，要小心点击。可以通过单击窗口左边的序列名称来选择单个序列，通过单击序列比对上方的数字标识选择序列的区域。选中的行用红框显示。

与其他任何菜单界面一样，File 菜单包括导入和导出选项。序列的比对可以从一个文件、甚至一个网址读入。Edit菜单包括删除、复制、移动选定序列或序列栏的命令。可以手工操作比对。可以在比对中单击任何一个字母来打开一个空位，用鼠标从任意方向把序列拖到该字母的左侧。 Colour菜单包括彩色标记比对选项，它们中的大多数包含最丰富的蛋白质序列信息。 Calculate 菜单包括计算一致序列和系统发育树的选项。

序列标志

目前非常流行的一种查看序列比对结果的方法是美国国家癌症中心（National Cancer Institute）的 Tom Schneider 开发的序列标志（logo）格式。这种格式对蛋白质基序等短序列区域非常有效。一致序列表示比对中的每个位置出现最常见的残基。比对中的其他信息，如在某一位置是否有其他残基出现，出现的频率是多少，在一致序列中不能反映出来。

序列标志，如图 8-1 所示，是用图形的方式来表示比对中每一位点的相对频率、信息内容、替代优先级的顺序和其他一些性质。

在一个序列标志中，序列每一位置的那一列字母代表的一致序列比单一字母的一致序列更详细。列的总高度代表在那个序列位置所包含的信息量（注 2)。单个字母的大小代表它们出现的频率。

注 2: 关于序列标志和生物序列数据中的信息的完整讨论，可以从National Cancer Institute 的 Tom Schneider 博士的网站下载详细的说明，其网址是：*http://www-lecb.ncifcrf.gov/~toms/index.html*。

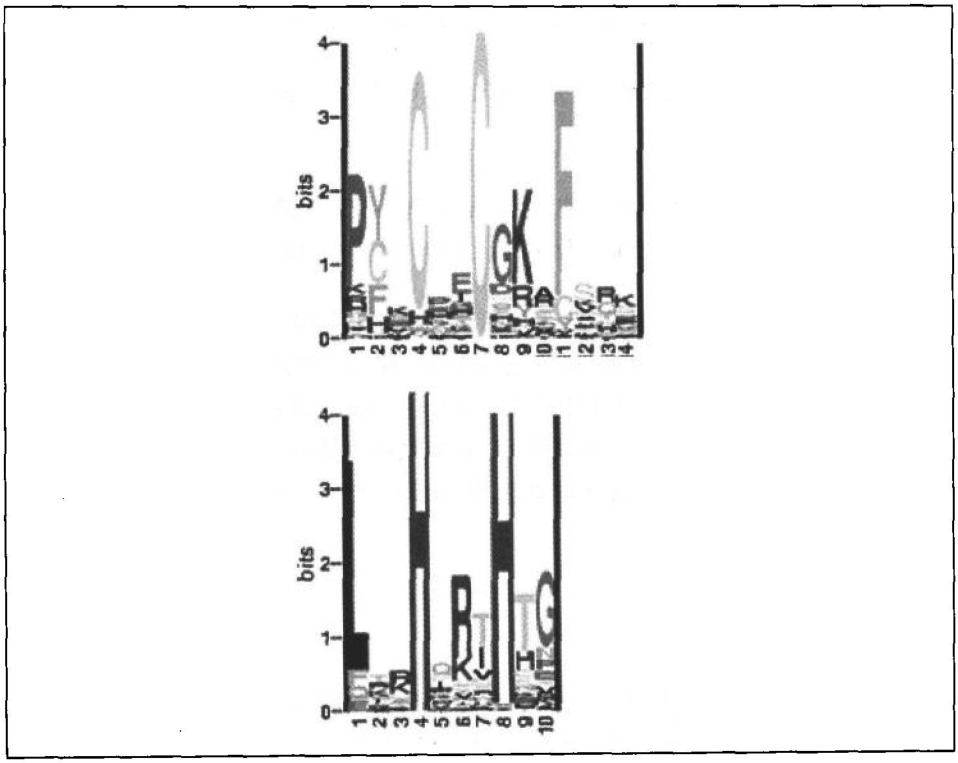


图 8-1　　序列标志

创建序列标志的软件是DELILA软件包的一部分，最初是用Pascal语言编写的。从比对好的序列创建标志，实际只需众多 DELILA 程序中的两个 ——*alpro* 和 *makelogo*。Pascal编译器通常不在Linux系统的编译器中，但如果你觉得这样比较冒险的话，可以下载一个标准的GNU编译器。编译这个软件的其他方法是使用这个程序的C语言版本。因为这些程序是从Pascal语言自动翻译过来的，所以需要在系统中安装 p2c（Pascal to C）库。

对初学者来说，一个更简单的方法是使用剑桥大学的序列标志Web服务器，这实际上是 DELILA 程序的作者推荐的。因此，我们认为它可以很好地执行预期的任务。比对序列可以用 ClustalX 生成的 FASTA 比对格式提交给该服务器。

系统发育分析

了解了一些多序列比对的基础知识以后，现在我们将介绍它的应用之一：系统发育推断。系统发育推断就是通过生物间可观测的性质来建立它们之间进化关系假说的方法。系统发育分析传统上是以物种的大体解剖学为基础的。在 Linneaus 建立界、门、属、种的分类系统时，早期的生物学家把生物分成图 1-3 所示的用符号表示的生命树。这种代表物种间关系的树状表示法，就是系统发育树，它已成为以序列相似性为基础表示进化关系的一种方便的示意图。序列关系的定量性质为开发更精确的树绘制方法和法则提供了可能。

尽管手工绘制的“生命之树”可能会按照本质上是艺术家对进化关系概念的理解而凭空进行分支，但是现代的系统发育树是严格的二叉分支：也就是说，在任意分支点，主分支仅分裂为两个子分支。二叉分支的树可以与其他任何分支模式类似。二叉分支树的假设极大地简化了建树算法。

在一个定量的系统发育树中，分支的长度可以不止有一种测定方式。在一组输入数据中，序列对相对于其他序列的进化距离是一种定义分支长度的方法。

假设所有的物种都有一个共同的特定祖先，物种的发展史通常都是有一个根（root）的，但是从序列数据得到的系统发育树可能有“有根（rooted)”和“无根(unrooted)”两种。在一组中计算任意两个序列的相似性和决定分支点的位置归属并不太难。而准确查明一个树中的哪一个序列有共同祖先，或者哪一对序列是这个祖先的第一个子分支则困难得多。虽然有些系统发育推断程序的确提出了树根的假设，但是大多数程序只能产生无根的系统发育树。图 8-2 和图 8-3 分别为有根和无根的系统发育树。

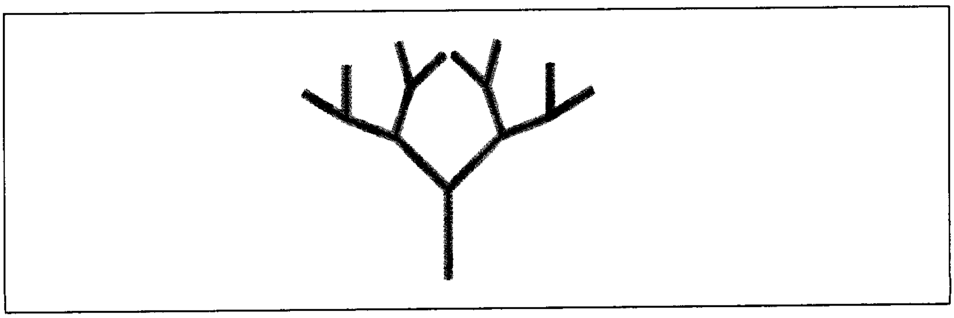


图 8-2　有根的系统发育树

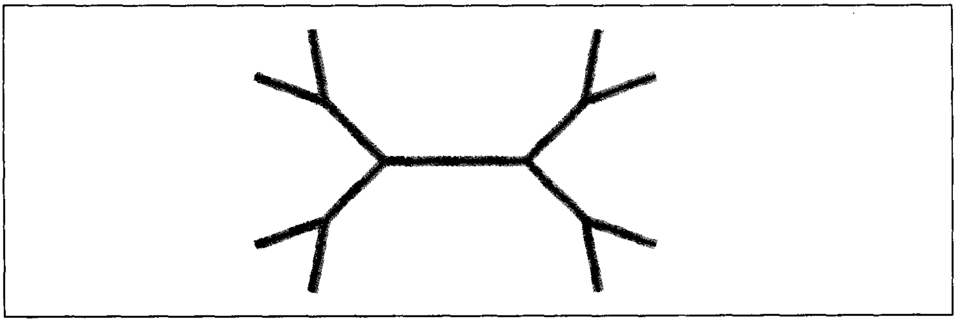


图 8-3　无根的系统发育树

从蛋白质或核酸序列推断出的系统发育史仅仅是代表整个真实生物界物种生成事件的生命之树的类似物，一个系统发育史可能是一棵树，可以描述一定的生物学本质，但单一的进化分析远远不能得出有关整个生物界的进化结论。以序列为基础的系统发育史是定量的。当这些结果是建立在足够的数据基础上时，它们可以为进化史理论提供有价值的科学证据。然而，以序列为基础的单个系统发育分析只能定量描述输入的数据集。作为一种定量工具，若超出其数据集的范围则是无效的。利用系统发育分析来建立进化假说时，必须牢牢记住这一点。

通过对不同蛋白质和基因家族产生的系统发育史进行比较分析，可以发现一种蛋白质可能比其他蛋白质进化得更快，某种蛋白质在一些生物中可能比在其他生物中进化得更快。因此，序列家族的系统发育分析对于某个特定基因的进化是最有信息价值的。整个生物界的系统发育理论只有通过对更大量的数据集的分析才能得到。

建立在成对距离基础上的系统发育树

成对距离法（pairwise distance method）是最容易理解的建树算法之一，该方法可以生成有根的树。这个算法首先通过在输入集中定义每一对序列间的距离矩阵初始化。然后序列按照距离分组，实际上就建立起了从树枝到树根的树。

距离的定义不止有一种方法，但DNA序列间差异性测量比较常见和简单的方法之一是 Jukes-Cantor距离，它与比对中两序列的有差异位点所占分数成对数相关。在两个不相关 DNA序列的无空位比对中，可匹配部分约占 25%，因此，测量到的 Jukes-Cantor距离接近无穷大，而不匹配残基比例接近 75%。

用于建树的成对聚类程序（ UPGMA ，使用算术平均数的未加权对群法）是直观的。首先每个序列被分配到自己的聚类中，从树的零高度开始这个序列分支（或端点）。无论根据什么距离测定方法，距离最近的两个聚类被合并为一个聚类。分支点（或称为节点）定义为连接两个分支的地方。节点在树中所处的高度反映了其所连接的两个端点的距离。这个过程反复进行，直到仅剩下两个聚类为止。当它们连接起来时就产生了树根。从理论上讲，该法所建之树的分支长度反映了进化时间的长短。

建立在相邻连接基础上的系统发育树

相邻连接(neighbor joining）是另一种距离矩阵方法，它消除了使用 UPGMA 方法可能产生的错误。在 UPGMA法所建之树中，绝对距离最近的分支处于相邻的位置。这个假设对树的拓扑结构的限制使其在某些情况下可能产生错误的树结构。

为了解决这个问题，相邻连接算法不仅会根据距离矩阵搜索最小的成对距离，而且会搜索使整个树高最小的相邻集。相邻连接是应用最广泛的距离建树方法，可以产生有意义的进化树，尤其适用于进化距离较短的情况。

建立在最大节约法基础上的系统发育树

另一个应用更为广泛的建树算法称为最大节约法。节约（ parsimony)一词与 Occam 的 Razor 原则有关，它是中世纪哲学家 William of Ockham 提出的原理：最简单的解释也许恰恰是正确的（注 3)。节约法是从一系列可能的树中找到一个需要最少的核苷酸或氨基酸替换就可以解释所看到的序列差异的树。

在比对序列的节约分析中，所关心的仅仅是那些可以提供进化信息的位点 —— 即那些喜欢选择树状拓扑结构而不是其他的位点。如果在一个位点不止有一个残基，并且每一类残基在比对中代表不止一个序列，这个位点被认为是有信息价值的。因此，对每一种可能的树的拓扑结构，每一位点所推论的进化改变的值是可以计算的。具有最大节约的拓扑结构所有包含信息的位点的进化改变值的总和最小。在有些时候，可能有多个树的拓扑结构有相同的节约性。

随着序列数的增加，可能的树的拓扑结构数也在增加。到达某个临界点时，完全计

注 3： 换言之就是，如果能事半功倍，就不要徒劳地事倍功半。

算每个拓扑结构的得分是不可能的。在这种情况下，寻找最大节约的简化算法是分支和边界（branch-and-bound）算法。这种算法通过一种快速或武断的计算，建立某个进化树的进化改变值的上限。在评价其他进化树时，超过这个上限值的结构在计算结束之前会被抛弃。

建立在最大似然法上的系统发育树

最大似然法（maximum likelihood method）也是对给定的一组序列的每种拓扑结构树进行评价。最大似然法是一种概率法。也就是，它通过在信息位点的每种可能的进化改变的概率排列并使树的总概率最大化来寻找最佳选择。最大似然法使用氨基酸或核苷酸的替换率信息，类似于多序列比对中的替换矩阵。

系统发育分析的软件

许多操作系统中都有系统发育分析软件的变体，在这么多选择中，应该使用哪种软件呢？目前可以得到的最为齐全的列表之一是由 Joe Felsenstein博士管理的，他是 PHYLIP 软件包的作者，该软件包可从 PHYLIP 网页上得到（*http://evolution.genetics.washington.edu/phylip.html*）。如果不想按照我们的例子使用 PHYLIP，可以很容易找到其他软件包的信息。

PHYLIP

最广泛发布的系统发育分析软件包是PHYLIP。该软件包包括30个可以实现不同系统发育分析算法的程序。每个程序都可以用命令行独立运行。在默认状态下大多数程序会寻找称为 *infile* 的输入文件，并输出称为 *outfile* 的输出文件。除了通过命令行标志输入参数外，和使用 BLAST 一样，这些程序还有交互式文本界面进行信息提示。

下面是刚开始分析蛋白质和核苷酸序列数据时，最可能用到的 PHYLIP 程序：

PROTPARS

利用节约法从蛋白质序列输入进行系统发育史的推导。

PROTDIST

利用最大似然法从蛋白质序列输入计算进化距离矩阵。

DNAPARS

利用节约法从 DNA 序列输入进行系统发育史的推导。

DNAPENNY

利用分支和边界法找到一组序列的所有最大节约的系统发育树。

DNAML

利用最大似然法从 DNA 序列推导出系统发育史。

DNADIST

利用 Jokes-Cantor 距离或其他三个距离标准之一，从DNA 序列计算距离矩阵。

NEIGHBOR

利用成对聚类算法或相邻连接算法从距离矩阵数据推导系统发育史。

DRAWGRAM

根据系统发育推论程序的输出结果绘制有根的进化树。

DRAWTREE

根据系统发育推论程序的输出结果绘制无根的进化树。

CONSENSE

从一组系统发育结果中计算出一致的进化树。

RETREE

允许用户对进化树进行不基于数据的交互式操作。

PHYLIP 是一个很灵活的软件包，程序之间可以以多种方式组合使用。为了用 PHYLIP 分析一组蛋白质序列，可以：

1. 利用 PROTDIST 读取一个蛋白质多序列比对，创建一个距离矩阵。
2. 将距离矩阵输入 NEIGHBOR，利用相邻连接产生一个系统发育史。
3. 将系统发育史读入 DRAWTREE，产生一个无根的系统发育树。

或者也可以:

1. 利用 PROTDIST 读取一个多序列比对，使用最大节约法产生一个系统发育史。
2. 利用 DRAWGRAM 读取系统发育史，产生一个有根的进化树。

每一个 PHYLIP 程序在 PHYLIP 软件包中以 *.doc 文件保存。这个文件已经被几个小组转化成了 HTML 格式。PHYLIP主页有基于 Web 的 PHYLIP 文档的链接。

几个组织在 Web上已经建成了可用的 PHYLIP服务器。在 SDSC生物学工作台的 PHYLIP版本可以产生可下载的 PostScript 输出。

PHYLIP 的输入格式。PHYLIP的分子序列分析程序可以接受比对（交叉存取的）格式的序列数据：

```
39
Archaeopt  CGATGCTTAC CGCCGATGCT
Hesperorni CGTTACTCGT TGTCGTTACT
Baluchithe TAATGTTAAT TGTTAATGTT
B. virgini TAATGTTCGT TGTTAATGTT
Brontosaur CAAAACCCAT CATCAAAACC
B.subtilis GGCAGCCAAT CACGGCAGCC

TACCGCCGAT GCTTACCGC
CGTTGTCGTT ACTCGTTGT
AATTGTTAAT GTTAATTGT
CGTTGTTAAT GTTCGTTGT
CATCATCAAA ACCCATCAT
AATCACGGCA GCCAATCAC
```

其中前十个字符是比对格式中序列的名字，其他行接在后面。在一个序列格式中，先是第一个完整的序列，然后是第二个完整的序列，依此类推。然而，在上述每一种情况下，序列必须由另一个程序预先比对。PHYLIP没有计算多序列比对的功能。

如果分析 PHYLIP的系统发育史输出结果，你会发现它是用嵌套的小括号中的代表每个序列的代码表示的。这称为Newick表示法。小括号的模式代表进化树的拓扑结构。最里面的小括号是树的最末梢的分支，如（A，B)。接下来的每一个小括号都连着另一对分支，如（(A，B)，(C，D))。如果产生系统发育史的算法可生成分支的长度，在 Newick表示法中，这些分支的长度可以明确地和每一个分支联系在一起，如：*((A:1.2,B:1.5):1.0,(C:2.5,D:0.5):1.2)*。

利用 ClustalX产生 PHYLIP 的输入数据

在本章前面讨论过的多序列比对程序 ClustalX使用相邻连接法绘制系统发育树。但是，可能更重要的是，它能从各种输入格式中读取序列，然后仅需使用 ClustalX界面中的 Save as （另存为）命令便可以从多序列比对中输出 PHYLIP 格式文件。

简图和基序

除了研究序列之间的相互关系外，多序列比对最成功的应用之一是发现新的相关序列。这种基于简图（profile）或基序的分析使用多序列比对得到的信息构建或寻找序列模式。在本节中，我们首先介绍有关基序的一些概念，然后介绍用这些原理搜索序列数据库的工具。

首先回顾一下，多序列比对是指从三个序列到上百个序列的比对。根据目的的不同，多序列比对可以跨越相关蛋白质的整个序列，也可以仅仅是相似区域。如图 8-4 所示的多序列比对通常是通过不断重复的序列和序列组间的成对比较，而不是通过直接的多序列比对而建立的。

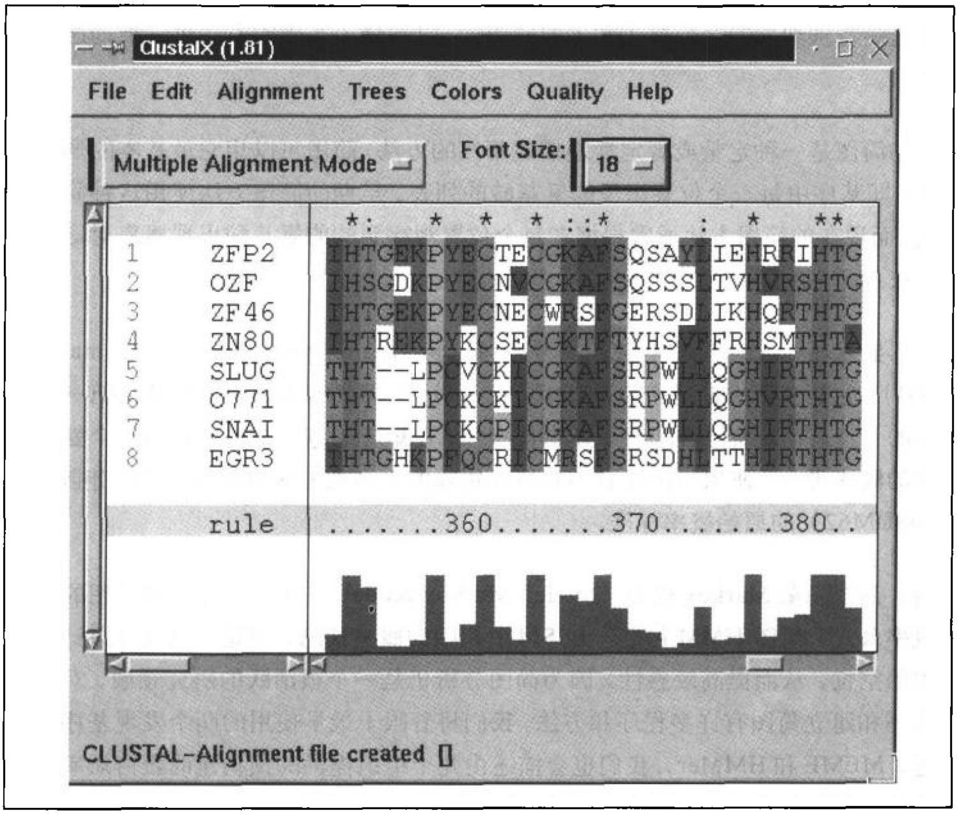


图 8-4　使用 ClustalX 的多序列比对

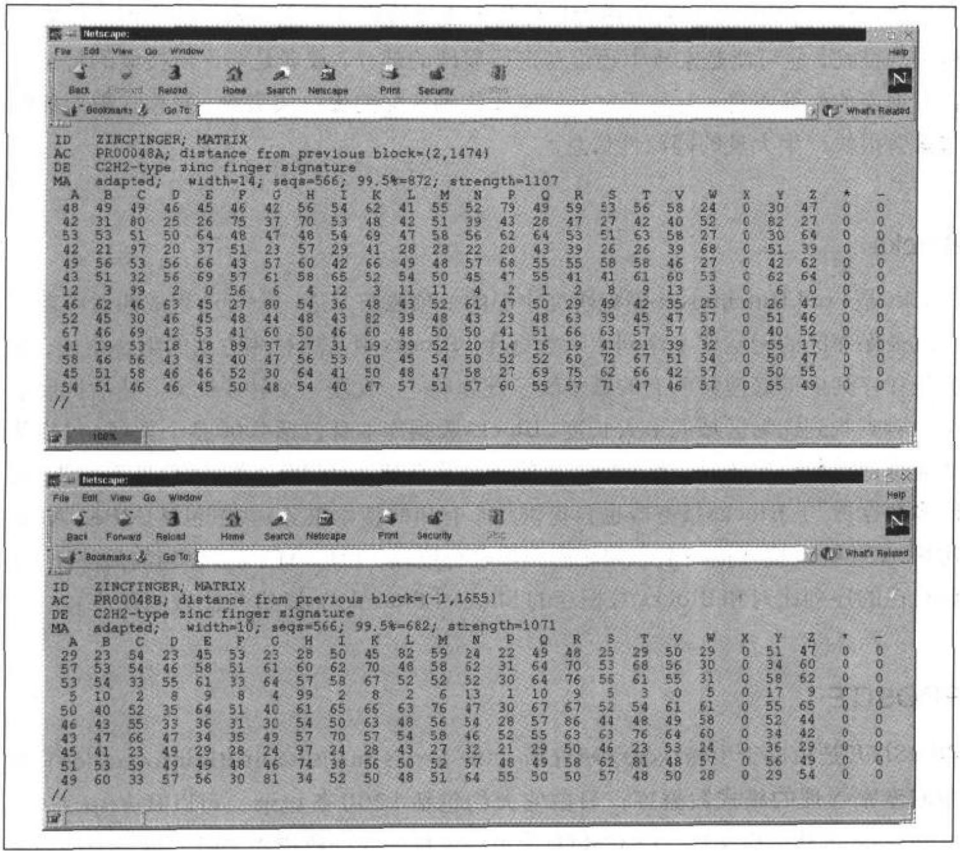
序列的基序(motif)是指序列中局部的保守区域，或者是一组序列中共有的一小段序列模式。基序这个术语更多的时候是指有可能具有分子功能、结构性质或家族成员相关的任何序列模式。基序在蛋白质、DNA、RNA 序列中都存在。但是基于基序分析的最普通的功能是在蛋白质中检测与结构或功能相关的基序。基序产生于多序列比对，可以以氨基酸的形式显示（如在 Prosite数据库中），或者以序列标志的形式显示。为了计算，它们可以表示为可以灵活变化的模式、与位置有关的得分矩阵或者简图隐藏 Markov 模型。

基序可以为蛋白质家族（protein family）或者一组进化相关的蛋白质而创建。蛋白质家族可以包括从非常相似到相当不同的许多蛋白质。尽管蛋白质家族的思想是一个普通的概念，但是选择蛋白质家族的方法和定义它的界限都依赖于研究者对它本身的界定。在成对序列的比较中有一个最低边界，超过它的同源性不容易检测到。通过检测特别细微的序列模式和远的同源性，以基序为基础的方法可以推动这个较低界限。

序列的简图是一种定量或者定性地描述基序的方法。简图可以用它最基本的形式来表现，即基序中每一个位置出现的氨基酸的列表。早期的简图方法使用这种简单的方式，而现在的简图方法通常根据在每个位置观察到的的氨基酸出现概率对氨基酸进行权重。

图 8-5 显示了一个与位置有关的得分矩阵（position-specific scoring matrix，PSSM)，它是一个代表基序的得分矩阵。不同于标准的得分矩阵，矩阵的第一维是基序的长度，第二维包括 20种氨基酸的概率。在矩阵的每一个位置是每一个氨基酸出现的概率得分。使用与位置有关的得分矩阵的大多数方法将标准化得分矩阵（如 BLOSUM62）的原始概率标准。

最后，简图隐藏 Markov 模型（hidden Markov model，HMM）是序列简图的严格的概率公式。简图 HMM 包括与 PSSM 中相同的概率信息，可是，它也可以计算比对中的空位，从而提高敏感性。因为简图分析仍是一个很活跃的研究领域，对于发现基序和建立简图有许多程序和方法。我们将着眼于最早使用的两个发现基序的程序包，MEME和 HMMer，我们也会描述由几个组织提供的预构建的蛋白质家族基序的可扩展数据库，它们中的一些与 PSSM 或 HMM 有关。



ID ZINCFINGER; MATRIX
AC PR00048A; distance from previous block=(2,1474)
DE C2H2-type zinc finger signature
MA adapted; width=14; seqs=566; 99.5%=872; strength=1107

A	B	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	X	Y	Z	*	-
48	49	49	46	45	46	42	56	54	62	41	55	52	79	49	59	53	56	58	24	0	30	47	0	0
42	31	80	25	26	75	37	70	53	48	42	51	39	43	28	47	27	42	40	52	0	82	27	0	0
53	53	51	50	64	48	47	48	54	69	47	58	56	62	64	53	51	63	58	27	0	30	64	0	0
42	21	97	20	37	51	23	57	29	41	28	28	22	20	43	39	26	26	39	68	0	51	39	0	0
49	56	53	56	66	43	57	60	42	66	49	48	57	68	55	55	58	58	46	27	0	42	62	0	0
43	51	32	56	69	57	61	58	65	52	54	50	45	47	55	41	41	61	60	53	0	62	64	0	0
12	3	99	2	0	56	6	4	12	3	11	11	4	2	1	2	8	9	13	3	0	6	0	0	0
46	62	46	63	45	27	80	54	36	48	43	52	61	47	50	29	49	42	35	25	0	26	47	0	0
52	45	30	46	45	48	44	48	43	82	39	48	43	29	48	63	39	45	47	57	0	51	46	0	0
67	46	69	42	53	41	60	50	46	60	48	50	50	41	51	66	63	57	57	28	0	40	52	0	0
41	19	53	18	18	89	37	27	31	19	39	52	20	14	16	19	41	21	39	32	0	55	17	0	0
58	46	56	43	43	40	47	56	53	60	45	54	50	52	52	60	72	67	51	54	0	50	47	0	0
45	51	58	46	46	52	30	64	41	50	48	47	58	27	69	75	62	66	42	57	0	50	55	0	0
54	51	46	46	45	50	48	54	40	67	57	51	57	60	55	57	71	47	46	57	0	55	49	0	0

//

ID ZINCFINGER; MATRIX
AC PR00048B; distance from previous block=(-1,1655)
DE C2H2-type zinc finger signature
MA adapted; width=10; seqs=566; 99.5%=682; strength=1071

A	B	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	X	Y	Z	*	-
29	23	54	23	45	53	23	28	50	45	82	59	24	22	49	48	25	29	50	29	0	51	47	0	0
57	53	54	46	58	51	61	60	62	70	48	58	62	31	64	70	53	68	56	30	0	34	60	0	0
53	54	33	55	61	33	64	57	58	67	52	52	52	30	64	76	56	61	55	31	0	58	62	0	0
5	10	2	8	9	8	4	99	2	8	2	6	13	1	10	9	5	3	0	5	0	17	9	0	0
50	40	52	35	64	51	40	61	65	66	63	76	47	30	67	67	52	54	61	61	0	55	65	0	0
46	43	55	33	36	31	30	54	50	63	48	56	54	28	57	86	44	48	49	58	0	52	44	0	0
43	47	66	47	34	35	49	57	70	57	54	58	46	52	55	63	63	76	64	60	0	34	42	0	0
45	41	23	49	29	28	24	97	24	28	43	27	32	21	29	50	46	23	53	24	0	36	29	0	0
51	53	59	49	49	48	46	74	38	56	50	52	57	48	49	58	62	81	48	57	0	56	49	0	0
49	60	33	57	56	30	81	34	52	50	48	51	64	55	50	50	57	48	50	28	0	29	54	0	0

//

图 8-5　常见锌指纹蛋白质序列基序的 PSSM

基序数据库

我们已经看到序列家族的简图和其他一致序列表示法可以用于序列数据库的搜索。因此，我们不必对使用一个序列进行搜索的基序数据库表示惊讶。基序数据库包括序列家族共有的保守序列的表示。今天，它们主要应用于注解未知序列。如果得到一个不在序列发生器中的新基因序列，在基序数据库中扫描它会得到它所编码蛋白质功能的第一手提示。

为了达到不同的目标，基序可以通过多种方法产生。有些依赖于自动分析，但是数据库往往需要专业管理者的大量手工劳动来维护。因为它们仅仅保存合理的大家族的基序，基序数据库相对于Genbank是很小的，它们不能反映蛋白质结构或序列数

据库的广度。应该意识到对基序数据库的不成功搜索并不意味着你的序列中没有可检测的模式。它可能是家族中还没有进行整理的部分，或者是与你所搜索的特定模式数据库的标准不符。对与所定义的家族的确匹配的蛋白质来说，对模式数据库的搜索会很快产生大量的同源性信息。

Blocks

Blocks是 Fred Hutchinson 癌症研究中心的一项服务，是一个自动生成的对应着蛋白质最保守区的无空位多序列比对数据库。 Blocks 由基序检测方法的组合生成，首先是在序列中彻底搜索所有的氨基酸三连密码子，发现一个种子比对，然后扩展这个比对到找到比对区域的最大长度。 Blocks 数据库本身包括4000 多个条目，包括从其他蛋白质数据库条目中 blocks 产生的条目共有超过10000 个条目(参见下一节“访问多数据库”)。Blocks 服务器也会提供几个有用的搜索服务，其中包括 IMPALA(使用 BLAST统计模型把一个序列和简图库进行比较）和 LAMA （多序列的局部比对；将自己的序列比对和 Blocks 数据库的 Shmuel Pietrokovski 进行比较的程序）。

PROSITE

PROSITE 是由瑞士生物信息学研究所（ the Swiss Institute of Bioinformatics）拥有的专家管理的模式数据库。目前它大约包括 1200 条记录，可以从 *http://ftp.expasy.ch* 下载平面文件。PROSITE 使用一个单一的一致模式来研究每一个序列家族的性质。PROSITE 没有开发结构化的以自动分析为基础的模式。取而代之的是，它们通过出版的原始文献或描述蛋白质特殊集团的功能的综述而被仔细挑选。在 PROSITE服务器上有一个滑稽的卡通来提示鉴定模式的最佳方法仅需要一个人、一支粉笔和一块黑板。PROSITE包括模式信息和能够检测新的模式实例的与位置有关的记分矩阵。

Pfam

Pfam是一个蛋白质结构域家族比对数据库。它由两个数据库组成：Pfam-A和Pfam-B。Pfam-A是含 2700 多个有空位的简图的可管理数据库，其中的大多数覆盖整个蛋白质结构域；Pfam-B 的条目是从产生 Pfam-A 的剩余序列中使用聚类方法自动生成的。Pfam-A 的条目起始于一个种子比对（ seed alignment)，管理员确信该比对具有生物学意义，并且可能会通过手工进行编辑。序列比对从每一个种子比对构建

简图隐藏 Markov 模型，并用它来搜索可得到的蛋白质序列的非冗余数据库。一个家族的完整的比对是从种子比对和任何新的匹配序列中产生的。这个过程可以重复进行以产生更广的家族和检测更远的匹配。Pfam的条目用科学文献中的信息进行注解，并且在可能的地方合并结构数据。作为最终的记录，Pfam 是 GeneWise genefinder 使用的简图 HMM 数据库，可用于搜索开放阅读框。

PRINTS

PRINTS是一个类似于 PROSITE的蛋白质基序数据库，但是它使用不止一个模式的指纹来研究整个蛋白质序列的性质。相对于整个蛋白质序列来说，基序通常是很短的。在 PRINTS 中，在序列家族中发现的基序组可以定义为该家族的签名。

COG

NCBI的 COG（Clusters of Orthologous Groups）数据库是不同类型的模式数据库。COG 通过比较 21 个完整基因组编码的所有蛋白质序列而建成。每一个聚类必须包括至少来源于三个独立基因组的蛋白质序列。COG的前提是蛋白质在许多不同生物的基因组中是保守的，这些蛋白质代表在整个进化中保守的原始功能。COG条目可以从 NCBI的 Web 网站的生物或功能目录访问到。COG 目前含有 2100 多条条目。

访问多数据库

那么，应该选择哪个基序数据库来分析一个新序列呢？因为比较完成得非常高效迅速，所以我们建议使用尽可能多的数据库，保存每一个数据库的最佳匹配、它们的得分和（如果可能的话）点击的意义。虽然 Blocks使用 InterPro 作为它自己模式的来源之一，到 2000年 6月它只包含无空位的模式，而不像 Pfam-A和 PROSITE 中那样包含有空位的简图。幸运的是，在此讨论的所有基序数据库都具有 Web 搜索界面，其中大多数可以接受 FASTA 格式或 FASTA 比对格式的输入。

一个允许整合许多基序数据库搜索的服务器是欧洲生物信息学研究所的 Integrated Resource of Protein Domains and Functional sites（InterPro是它的助手）。InterPro 允许将序列与来自 Pfam、PRINTS、ProDom和 PROSITE 等的所有基序进行比较。InerPro 的基序按照源蛋白质的名字、基序发生的蛋白质实例、参考文献和相关基序进行注释。

构建和使用自己的简图

如果寻找已有完备资料的蛋白质家族，基序数据库是有用的。但是，如果想通过 Genbank 发现你要用的新基序，或者想创新并在不通常的地方寻找模式，那么你需要建立自己的简图。有几个软件包和服务器可以用于基序发现（motif discovery），从一组序列中发现和构建自己的基序的程序。构建基序的最简单方法是从一个多序列比对中寻找保守区域。虽然通常我们建议使用自动方法，而非手工方法：自动化可使你的工作更快，更有可重复性，更少犯错误。除了 Block Maker 外，还有许多其他程序常用来搜索和发现基序。在本节中，我们将介绍 MEME 和 HMMer 两个程序的使用，这是两个常用的基序分析软件包。

在开始之前，先看两个有关基序发现的观点。第一个是随着 IntePro 和 Blocks 的增长，通过它们的成员数据库之一发现全新的没有备案的序列基序变得越来越困难。一定要通过在数据库中搜索序列或者使用 Blocks 服务器的 LAMA 程序这类基序比较工具，在一些已知的基序中检查你的基序。第二个是为了发现可靠的模式，并使用它们进行检索，需要许多序列。我们曾经在含有很少（5~10）序列的方案中使用过这些程序，但是，作为一种经验，要得到可靠的基序预测，需要 20个以上的序列。序列越多得到的基序越可靠。

使用 MEME 发现新基序

MEME程序是由圣地亚哥加州大学的 Charles Elkan、Tim Bailey和 William Grundy 开发的一套基序分析工具。MEME是基序启发（Motif Elicitation）的多 EM的缩写[而 EM是期望值最大化（Expectation Maximization）的缩写，是统计学中预测丢失值或未观察到的值的方法]。它们可以在网上使用（*http://meme.sdsc.edu*），也可以下载它们的 C 源代码在本地计算机上编译和运行。在此，我们看一下网络版。在 MEME 程序组中共有三个程序：

MEME

在一组未比对的序列中发现共有的基序。

MAST

提取 MEME 发现的基序，用它搜索一个序列数据库。

MetaMEME

从多个 MEME 基序中构建模型，并用它搜索数据库。

在将一组序列提交给 MEME时，同时也在测试一个假说，虽然你不知道整个的序列比对，但是它们有共同的短的序列相似区域。通过在 Web 表单上输入 email 地址和一组需要寻找基序的序列，可以开始使用 MEME 。序列的格式可以是几种格式中的一种，虽然最常用的是 FASTA。在提交页的底部是需要设置的一些参数，包括希望每个序列中基序出现的次数，希望找到基序的数目和每个基序的大体宽度。

结果通过三封邮件返回给你。第一封仅是确认消息，告诉你工作正在进行中，第二封（邮件主题是“ MEME 工作 xxxxx 结果: ”,其中 xxxxx是 MEME服务器给工作的编号）用人工和机器可读形式给出 MEME 的基序预测结果。这是你搜索数据库所需的消息，一定要将这个消息的内容储存为文本文件，以便以后可以将它提交给 MAST或 MetaMEME。第三封（主题是MEME 工作……MAST分析：)是一个 HTML 文件（可通过 Web 浏览器查看），它会显示每个基序在所提交序列中的位置。每条消息都已经存档并有详细的内容解释。

利用 MAST 和 MetaMEME 搜索基序

基序分析的下一步是看在其他序列中是否存在你的新基序。 MEME服务器提供了两个独立的程序 —— MAST 和 MetaMEME，它们允许使用新 MEME 基序在一个数据库中进行搜索。MAST 仅仅搜索每个基序并报告匹配的序列，而 MetaMEME 将多个 MEME 基序结合成隐藏 Markov 模型，用这个模型来搜索数据库。MAST和 MetaMEME都将第二封邮件中的MEME基序预测结果作为输入（注 4）, MetaMEME 也可使用在创建 HMM时产生 MEME 基序的原始序列作为输入。两个搜索的结果可以显示每一个匹配的位置、得分和它的统计学显著性。

利用其他程序发现基序

与前面提到的一样，有许多程序可用来在一组未比对的序列中发现基序。除了我们提到的外，你还可以试试这些：圣克鲁斯加利福尼亚大学的 David Haussler 及其合作者开发的 SAM HMM 程序；斯坦福大学 Brutlag小组的 Emotif 和 Ematrix服务器；

注 4：　你的确像我们建议的那样将第二封邮件存为了文本文件，是吗？

以及可从 NCBI 下载的 ASSET、gibbs 和 Probe 等工具。还有，早期可做的一件事情是用 LAMA 程序对基序和 Blocks 数据库中的基序进行比较。如果你看上去的确发现了一个新基序，比较一个或多个其他基序发现工具的结果是有用的。如果所有的程序都从同样的序列中发现了同样的基序，你会对你的结果更有信心。

HMMer

HMMer 是一个建立简图 HMM 的软件包。HMMer 的中心功能位于 *hmmbuild* 程序和 *hmmcalibrate* 程序中，前者可以从一个序列比对中产生一个简图 HMM，后者可以校正搜索统计量。HMMer 软件包也包括根据 HMM 随机产生新序列的工具，以一个简图作为查询搜索序列数据库的工具，以一个序列查询搜索简图数据库的工具，下面列出一些易用的应用程序：

getseq

通过名字从大的一个平面文件数据库中提取序列。使用命令行从一个数据库中选择特殊的记录是很容易的。

hmmalign

读取序列文件和简图 HMM 并创建多序列比对。

hmmbuild

从多序列比对建立简图 HMM。它可以产生整个比对的整体结果或多个局部比对的结果。

hmmcalibrate

读取 HMM 并校正它的搜索统计量。

hmmconvert

将 HMM 转变成其他简图格式，特别是 GCG 简图格式。

hmmemit

基于简图 HMM 随机产生序列。它也可产生一致序列。

hmmfetch

如果所需要记录的名字已知，可从数据库中检索简图 HMM。

hmmindex

为简图 HMM 数据库作索引。

hmmpfam

用一个查询序列搜索简图 HMM数据库（如 Pfam）。如果你想注释一个未知序列，可以使用它。

hmmsearch

用简图 HMM搜索数据库。如果想在数据库中找到更多的模式实例可使用它。

sreformat

把一个序列或比对文件从一种格式转换为另一种格式。

HMMer 从包括 ClustalW 在内的几种不同的序列比对程序读取多序列比对文件。HMMer 作者推荐将 ClustalW 作为为 *hmmbuild* 产生输入文件的多序列比对工具。

HMMer 可以从华盛顿大学的 Sean Eddy 博士的网站下载（*http://hmmer.wustl.edu*）。HMMer 是一个行为良好的程序，源程序在 Linux系统上安装没有任何困难，仅仅按照安装文件的说明进行即可。它甚至可以安装自己的 Unix 手册页，以方便你使用 man 命令访问每个 HMMer 程序的在线帮助。每个 HMMer程序命令行选项的特殊信息通过使用 $-h$选项运行程序也可查看到。

将基序信息整合进成对比对

多序列信息可以优化成对比对。BLAST软件包具有两个新模型，它们可以使用多序列比对来提高数据库搜索的专一性。这些模型可以通过 blastpgp 程序访问。

PSI-BLAST（Position Specific Iterative BLAST，位置专一性迭代BLAST）是原BLAST 程序的强化版，它实现了可以提高数据库搜索专一性的简图。 PSI-BLAST 从单序列开始，利用有空位的 BLAST在数据库中寻找局部比对，创建和起始序列长度相同的多序列比对和简图。然后简图再次用于搜索蛋白质数据库，寻找局部比对。这个过程可以迭代多次。对使用 PSI-BLAST的一个忠告是你需要知道在哪里终止。比对中的错误可以通过迭代而变大，在最终的序列搜索中引起假阳性。 PSI-BLAST 可以通过运行 *blastpgp* 单独使用。然而，NCBI 的 PSI-BLAST服务器可能是运行PSI-BLAST搜索的最佳方式。服务器需要你确定一次迭代之后是否还要进行另一次迭代，你也可以在每一步手工挑选有利于简图的序列。

PHI-BLAST（Pattern Hit Initiated BLAST，模式命中起始 BLAST)使用待查序列和该序列中的预选模式作为蛋白质数据库的输入查询。模式必须以PROSITE语法表示,这在 PHI-BLAST服务器站点上有详细描述。PHI-BLAST 也可以起始运行一系列 PSI-BLAST 迭代，它可以是单独的程序或者是（对用户更加友好的） Web 服务器。

第九章

蛋白质结构的可视化和结构性质的计算

蛋白质三维结构分析是一个比生物序列分析更成熟的领域。20 世纪 70年代早期开始发布大分子晶体结构的坐标，从那时起，许多研究小组和公司开始开发显示和测量蛋白质结构性质的软件。

结构的可视化和结构性质的测量是分子和结构生物学家的重要工具。能够看到蛋白质的三维结构并详细分析它的形状可以对催化位点和相互作用位点的定位有所启发，并能够帮助鉴定定点突变研究的靶点，定点突变常常用于研究蛋白质功能化学的详细性质。

下面是一些这类方法在分子生物学中的最新应用：

- 山峰雪松花粉中的致敏蛋白分子模型的建立和随后对引起过敏反应的区域的确定
- HIV病毒中 DNA逆转录酶的突变活跃位点的性质；这个位点被认为与 HIV病毒的快速突变能力有关
- 与青春期综合症相关的 DNA 结合蛋白的模型的建立和引起该病突变的性质

在蛋白质结构文献中有许多专业的分析程序，我们不想列出所有的方法。而是，结合每一目的的软件实例，介绍蛋白质结构分析和建模的标准方法，这些软件的目的

有：可视化和绘图；几何和表面性质分析；分类；分子间相互作用和与溶剂的相互作用分析；一些物理化学性质的计算。

对所有目的的分子结构建模来说，最易用的工具还是MSI公司的Quanta和Insight，Tripos 公司的SYBYL和其他一些商业软件包。然而，要获得这些软件的使用许可，尤其是对于多用户来说，是非常昂贵的，并且它们往往需要指定的高端硬件设备（如SGI和 IBM 的 Unix 工作站）来运行。本章我们将再次把目光集中于在标准台式PC机的 Linux 上可以运行的软件或者可以在任何平台的 Web 浏览器中运行的软件。

关于蛋白质结构数据

因为蛋白质结构分析是一个相对较老的领域，在计算机应用历史中它比序列分析发展得更早，所以也继承下来了一些不便之处。虽然许多软件都使用标准的PDB格式，但是其他一些软件，尤其是一些分子模拟软件，则需要使用一些稍微不同或显著不同的输入格式。因为蛋白质结构分析软件都比较老，许多程序是用 FORTRAN语言编写的，对数据输入的格式非常挑剔。PDB对数据的标准化非常好，但是对个别软件包水平的标准化就不太好了。如果要对蛋白质的结构数据进行大量的处理工作，必须学会在需要的时候用一些程序把结构文件转换成合适的格式。第十二章我们将演示一个简单的结构文件格式转换的例子。

Brookhaven PDB格式是大多数结构分析程序使用的蛋白质结构数据格式。在 20世纪70年代这种格式迎合蛋白质结构领域的需要而产生，尤其是这种格式可以人工阅读，也可以利用 FORTRAN程序进行编译，因为它使用有严格结构的 80个字符行。这种格式包括一个首部，包含各种有关结构的信息，如文献引用；分辨率；结晶学参数；序列；有时还有二级结构信息；另一部分包含原子记录。由 *ATOM* 标记的原子是蛋白质链的一部分，而由 *HETATM* （杂原子集团）标记的是辅助因子分子、底物、离子或者其他与蛋白质链非共价结合的集团的一部分。关于 Brookhaven格式的详细逐行解释可在 RCSB PDB 的网站得到。

蛋白质结构文件也可以从 PDB 网站以一种叫做 mmCIF（the Macromolecular Crystallograghic Information Format，大分子晶体信息格式）的新格式得到，或者从 NCBI以 ASN.1文件格式得到。这两种格式可被计算机高度解析，如果编写用于

分析蛋白质结构的计算机程序，它们可能比老式的 Brookhaven 格式更好用。然而，你也必须考虑到用户群仍然和 Brookhaven 格式联系在一起。

蛋白质的化学性质

要运用蛋白质的序列和结构，需要了解一些蛋白质的化学性质的知识 —— 你可能从大学的有机化学教程中得到的知识。在这里我们将提供一些有关词汇，你可以从参考文献中得到更多的相关知识。如果你已经知道需要的蛋白质的化学性质的知识，你可以跳过这一节，从下一节开始。

研究蛋白质结构时，需要有一些基本的有机化学知识，其理由是很简单的。蛋白质经常通过氨基酸和与蛋白质结合的有机小分子（辅助因子），或金属离子的介导，以标准的有机化学反应机制来行使蛋白质的功能。为了理解蛋白质结构可能是如何催化反应的，需要对有机反应的机制有足够的了解来建立假说：对于给定的蛋白质形状和各种氨基酸的位置，反应是如何进行的。

即使在催化机制不是你关心的主要问题的情况下，化学性质也会起重要作用。蛋白质的结合常常是通过蛋白质结构的静电性质介导的，相互作用的分子可以通过很强的静电位从相当远的距离拖到一起。在蛋白质结构内，氢键和其他原子间的相互作用对结构的稳定起着重要作用。原子间的相互作用和分子形状是分子间专一性相互作用的基础，分子间的相互作用包括蛋白质与其他蛋白质或小分子底物的相互作用。你可能关心在生物化学实际应用中的分子专一性 —— 设计小分子或多肽药物，了解疾病和免疫的分子基础，或者对参与细胞间和整个机体中传递分子信号的特殊分子进行深入研究。

本章中的工具能使我们看到蛋白质的结构、看到它的特征是什么、定位不同类型的氨基酸和观看蛋白质的特殊子集、测量距离和表面积、计算溶剂可接近性和静电位等各种空间变量的性质。然而，使用这些工具所能做的工作依赖于我们对蛋白质化学性质的理解。

从一维到三维

蛋白质的化学性质是如何与其一维序列相关的呢？在第八章中，我们讨论了在蛋白

质家族中发现具有保守特征的模式（基序）的技术。我们可以在一维数据中发现这些序列模式，因为虽然蛋白质的三维结构是复杂的，但是它是由组成蛋白质的氨基酸的固定顺序以某种方式决定的。在序列中保守的基序往往与蛋白质家族的重要结构和功能特征有关。这些特征往往可以通过它们在蛋白质结构中扮演的角色而被理解。

当氨基酸在序列中连在一起形成多聚体时，它们是通过在每个氨基酸的碱性氨基集团和酸性羧基集团间形成肽键而连在一起的（见图 9-1）。这便形成了有重复骨架结构的氨基酸长链。

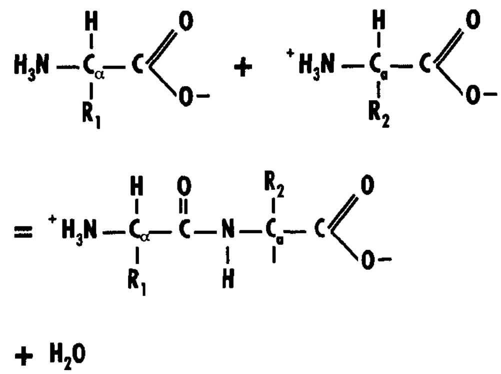


图 9-1　肽键、肽链（化学符号表示法）

每个氨基酸的可变集团从重复的骨架中伸出来，在蛋白质结构术语中被称为侧链（sidechain，见图 9-2）。20 个氨基酸侧链中的每一个在化学上的某些方面是不同的。

侧链可以以许多方式来分类。一些相对较大，一些相对较小，而有一个氨基酸没有侧链。有些带有正电荷或者负电荷，有些是油性的，或者说是疏水性的，表明它们在能量上是不利于溶于水。另外一些是亲水性的，易溶于水。有些有大的环状结构，而另一些只有直链碳链。

有些是酸性的，其他一些是碱性的。蛋白质中特殊位置的氨基酸在进化中是保守的，因为它是那个位置需要的，或者起到稳定蛋白质结构的作用，或者形成特殊的结合位点，或者催化一个反应。通过查看序列数据可以发现蛋白质中个别的氨基酸是保守的，但是要建立它们为何保守的假说，需要考察蛋白质的三维结构。图 9-3 显示

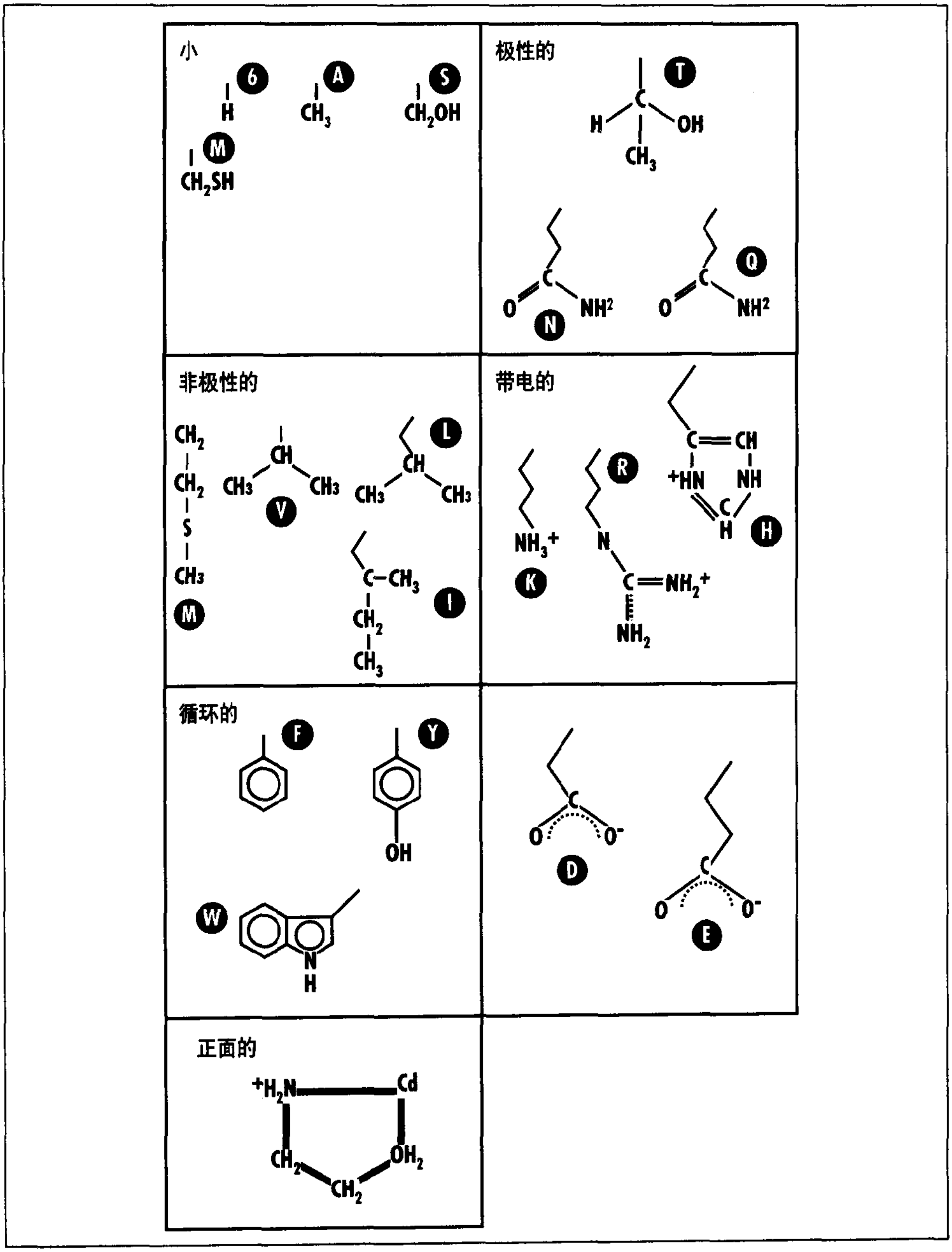


图 9-2　　氨基酸侧链（化学符号表示法）

了 20种氨基酸被分成化学上相似的组。注意许多氨基酸被分到了不止一个目录中。

氨基酸的侧链可以同时是"非极性"和"碱性"的。例如，赖氨酸，它有一个长的脂肪侧链，末端是一个氨基集团。因为在化学性质和氨基酸之间不是一对一的关系，而是多对多的关系，所以不能总是简单地预测一个氨基酸替换的效果。

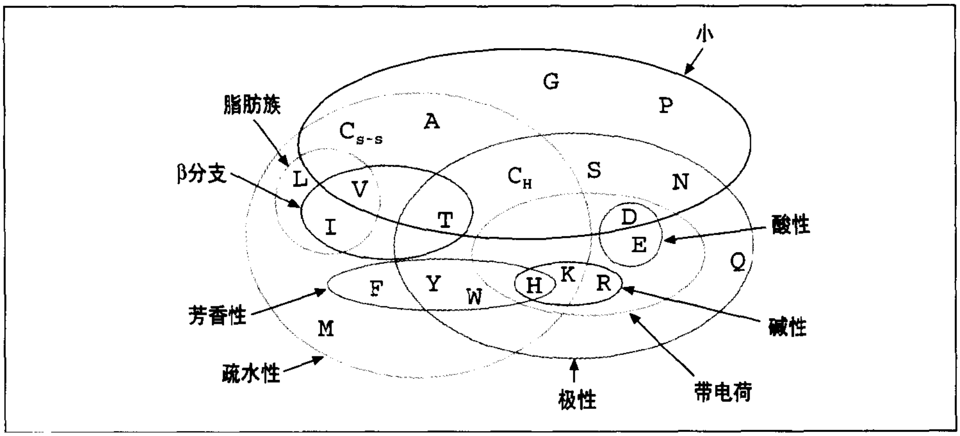


图9-3　　氨基酸侧链（Venn 图表示的分类）

原子间的相互作用力不仅对形成结合和作用位点的特殊相互作用起作用；它们也对蛋白质结构中总能看到的标准模式的形成起作用。氨基酸骨架在空间上按照一定的方式受到限制——因为原子间的相互碰撞而不能以某种方式移动。你可能早已熟悉在蛋白质结构中常常出现的 α 螺旋和 β 片层结构。两者经常出现的原因是蛋白质骨架中的空间限制。

从已知的氨基酸结构中，Pauling 和 Corey 首先预测出蛋白质结构中存在 α 螺旋和 β 片层。Ramachandran 首先精确描述了在肽链中氨基酸可以采用的构象的范围。肽链的构象可以用蛋白质骨架中二面角的值简单地描述（也就是：N-Cα键周围的四个原子所描述的角和 Cα-C键周围的四个原子所描述的角）。这些角分别称为 Φ和 Ψ。在蛋白质骨架中，链是不能绕着第三种键（肽键）自由转动的，因为它有部分双键的性质，因此在化学上被限制在一个平面内，所以每一个氨基酸的 Φ和 Ψ值就提供了蛋白质骨架的完整描述。Ramachandran 图仅是整个蛋白质结构的 Ψ 对 Φ 的散点图。评价蛋白质结构模型的一个方法是将它自己的 Ramachandran 图和 Φ 与 Ψ 允许值的普通 Ramachandran 图进行比较。

图 9-4 是一张普通的 Ramachandran 图，它显示了蛋白质结构中氨基酸被允许的 Φ 和 Ψ 值的组合。图的左下象限的小阴影区是 α 螺旋中氨基酸的标准构象，左上象限的大阴影区是 β 片层或其他结构中氨基酸的标准构象。

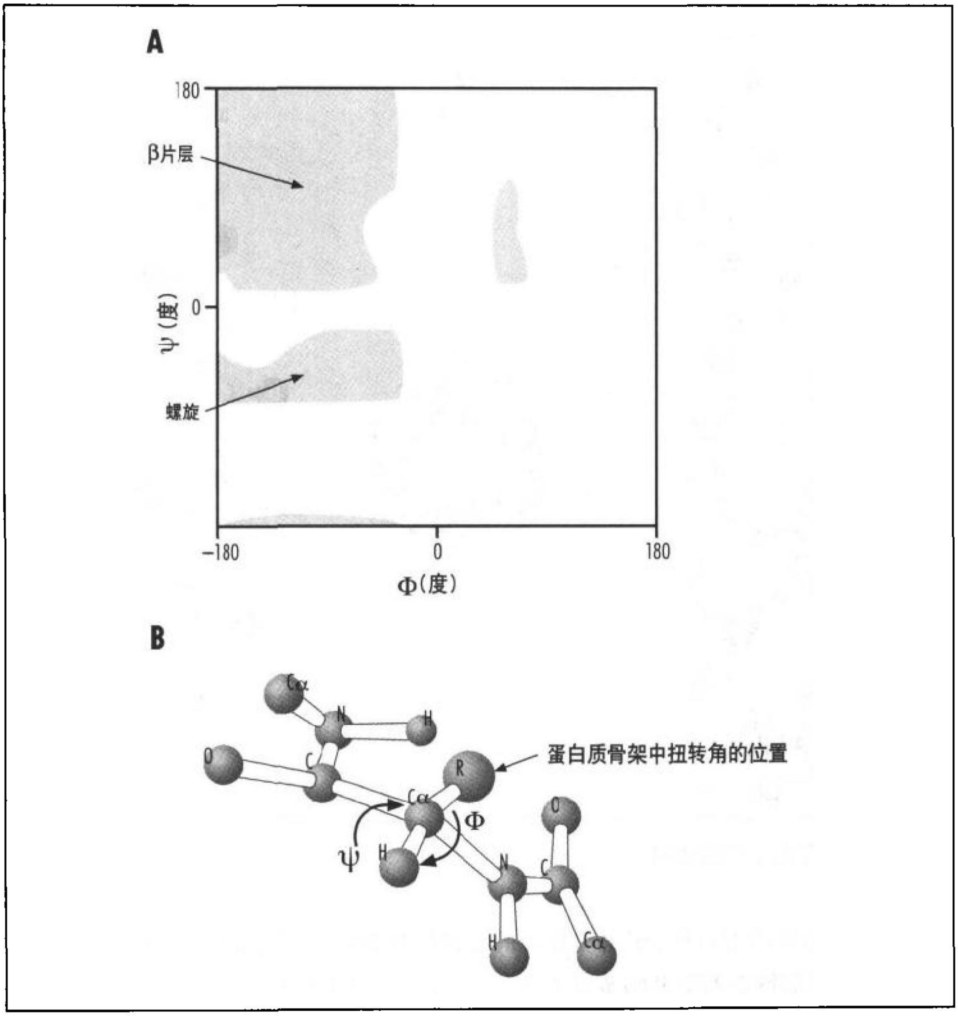


图9-4　蛋白质骨架允许构象的 Ramachandran 图

从 Ramachandran 图中可以明显看出空间相互作用是蛋白质结构一般特征的非常重要的决定因素。对于从一条线性氨基酸链形成紧密的结构的方式，空间相互作用可以除去蛋白质大部分的可能构象，留下相对较少的选择。

蛋白质的序列称为一级结构。蛋白质的基本组织层次是氨基酸序列。图 9-5 所示的 α 螺旋和 β 片层结构都称为二级结构，是下一个组织层次。多个二级结构组分间的相互作用产生超二级结构和三级结构 —— 螺旋和片层相互结合，形成更大的可以用拓扑学描述的特征结构。

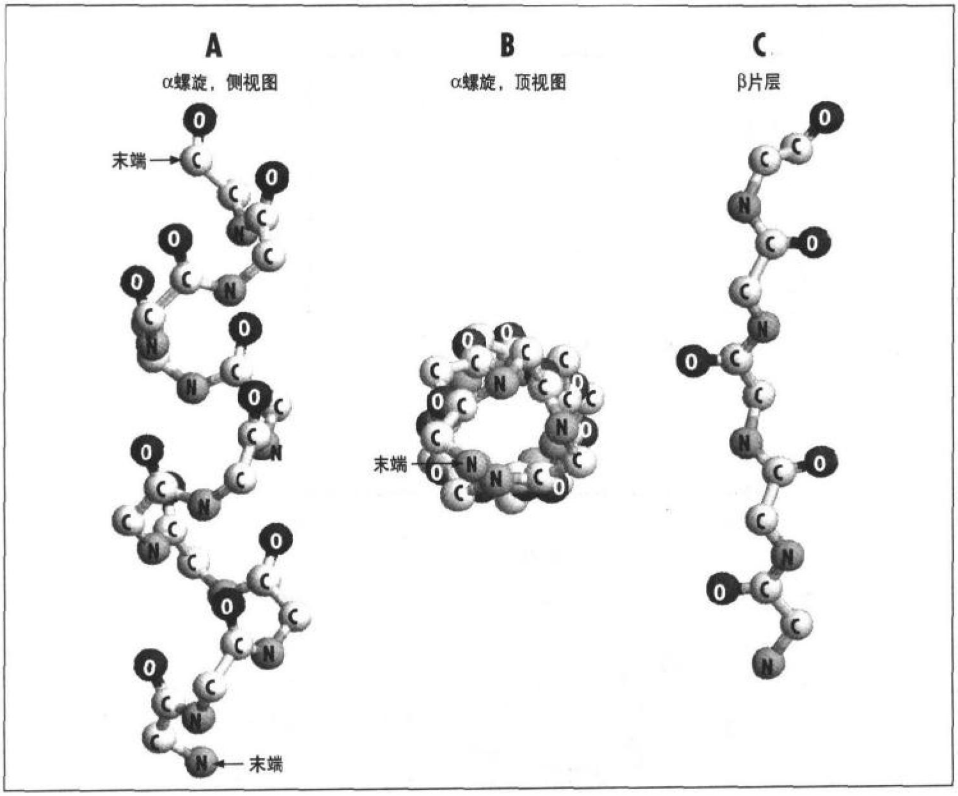


图 9-5　α 螺旋和 β 片层结构

为了产生有功能的蛋白质，蛋白质链中的氨基酸序列必须形成适合的蛋白质三维折叠，它也必须把特别的氨基酸放在合适的位点，使它们在这个框架中执行蛋白质化学。找到方法来从已知蛋白质序列中找出这些化学成分，用公式把它们表示为规则，用这些规则来预测其他蛋白质的结构是生物信息学中最大的开放研究问题之一。

相互作用力和蛋白质结构

因为蛋白质结构的形式和它的化学特征都是由原子间的相互作用控制的，所以对在蛋白质结构中扮演重要角色的原子间的相互作用，至少有一个基本的了解是很重要的。在物理学上原子间的相互作用是很复杂的，详细描述它需要另一整本书，很幸运已经有其他人写过了。你可以参见参考文献。我们希望给你的是这些力的最基本的知识，来帮助你理解为什么可以开发计算机方法来测量和计算蛋白质的结构性质。

理解这些力是我们设计评价和预测方法的基础。穿线方法（threading method）依赖于区别处于适宜化学环境中的氨基酸和非适宜环境中的氨基酸的能力。同源性建模和结构优化方法依赖于原子间隔、键长、键角和其他值的规则。这些规则可以从小分子的化学实验或从发布的已知蛋白质结构的观测值中得到。可是，虽然这些规则反映了原子间能量的相互作用，但是这些规则是人为构建的。

共价相互作用

共价相互作用发生在非常短的距离内（大约 1~1.5 埃）。它们是把原子结合成分子的非常强的作用力。在共价键中，有关原子实际是共享电子的。与蛋白质结构中的其他作用力不同，共价键实际上是在某种程度上改变了相关原子的属性，形成共价键的原子不再是离散的实体，而是结合在一起形成新的分子。

蛋白质骨架，包括连接氨基酸的肽键，都是通过共价键连接起来的。在蛋白质结构中氨基酸保留了一些它们各自的化学特性，但在形式上它们是新分子的一部分。单个氨基酸侧链的原子间也是共价键相互结合的。这些共价键对蛋白质结构中的原子间距离有很强的约束作用。

因为共价相互作用受到物理化学规则的强烈制约，结构质量确认过程的一个重要部分就是确认键长、键角和二面角没有强烈偏离它们的允许值。共价键的长度由相关原子的大小和类型，及原子间共享电子的数量决定。共享电子越多，键越短、越强。键角受到原子轨道的结构限制。二面角（dihedral angle，两对键合的原子相互间围绕中心键旋转的角）主要受制于空间位阻。这些化学限制也用于大分子模拟，在那里，它们与保持分子允许构象的实际作用力有关。

氢键

两个极性集团相互作用时可以形成氢键，但是两个极性集团必须是特殊类型。一个必须是质子供体，即质子（氢原子）共价结合到一个强的电负性原子上的化学集团，如氧原子。质子和电负性原子间的键是极性的，使得质子带部分正电荷，电负性原子带部分负电荷。另一个集团必须是质子受体，即一个带有部分负电荷而且没有结合质子的电负性原子。第一个集团中的带正电的极化的质子吸引带负电的极化的第二个集团，两者形成一个非共价键，但是虽然如此，氢键还是比一般的非共价键的相互作用更短更强。因为氢键有很强的方向性，这在非共价和静电相互作用中是不常见的；如果相关的三个原子形成的键角过大或过小，氢键会被减弱。

氢键相互作用是蛋白质结构中最重要的稳定作用力之一。蛋白骨架包含一个质子供体（在它的N-H集团中)，和一个质子受体（在它的羧基氧中），两者沿着肽链被有规律地间隔开（见图9-6）。这些集团的相互作用稳定了两种主要的二级结构，α螺旋和β片层结构（见图9-7）。因此，一些结构预测方法总是试图使用潜在的氢键对来提高预测的准确性。

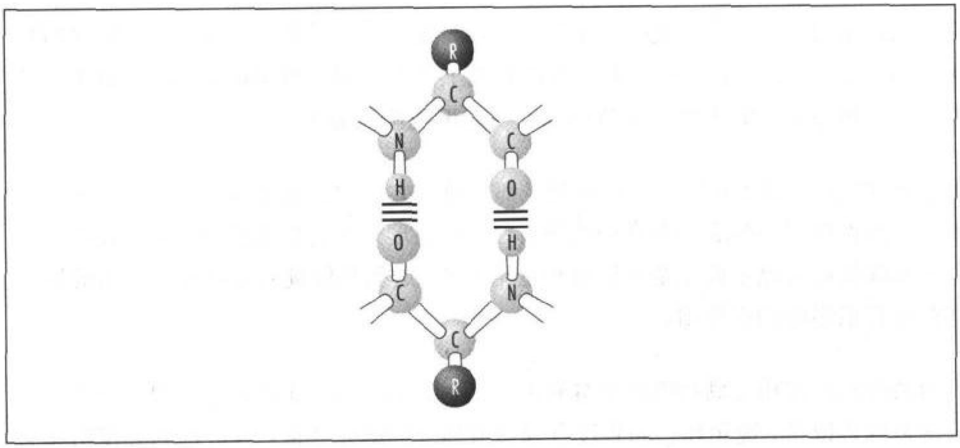


图9-6　蛋白质骨架中的质子供体和受体

疏水和亲水相互作用

在蛋白质结构中讨论最多的（和经常错误使用的）概念是疏水作用力。在过去我们已经提到，氨基酸可以分为疏水性的和亲水性的。但它们的确切意义是什么呢？

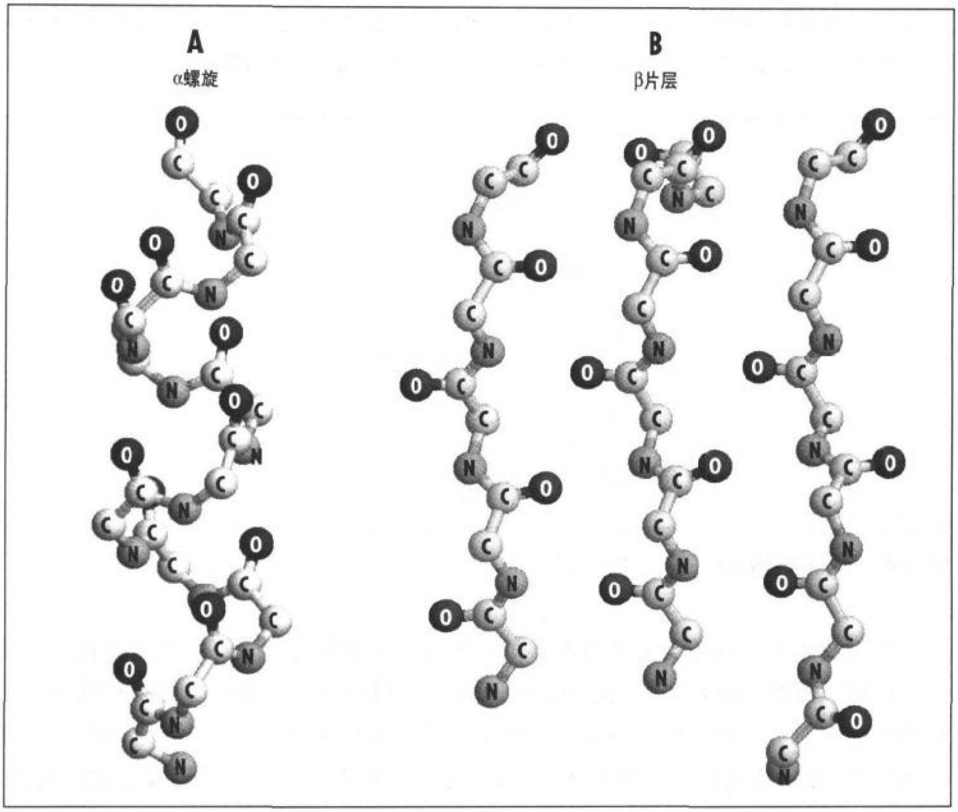


图 9-7　α 螺旋和 β 片层中的氢键

除了细胞膜中结合的蛋白质以外，蛋白质总是处于水溶液中。它们不断与水分子发生相互作用。水是一种有一些有趣性质的溶液。这些性质有利于细胞蛋白质特征性的紧密球状结构的稳定。

水是极性分子。液体水中的单个水分子能和相邻的水分子相互形成 4 个氢键。液体水是基本上不间断的氢键结合的分子组成的格子，如图 9-8 所示。这种性质有利于水的高熔点和高沸点，也有利于低压缩性和高表面张力的性质。这也导致了水与可溶性蛋白质之间的有趣的相互作用。

溶解于水中的非极性分子会打断液体水中有规律的氢键格子。单独的水分子可以重新定向到小的非极性分子周围，来保持它们的氢键网络，但是这种定向需要一种称为自由能的消耗（自由能是化学中消耗的度量）。非极性溶质的存在迫使水分子进入

一种比平时更有序的构象。除了不能自由转动和定向外，靠近非极性溶质表面的水分子还不得不包围它，并形成一种笼状结构。这在熵上是不利的。

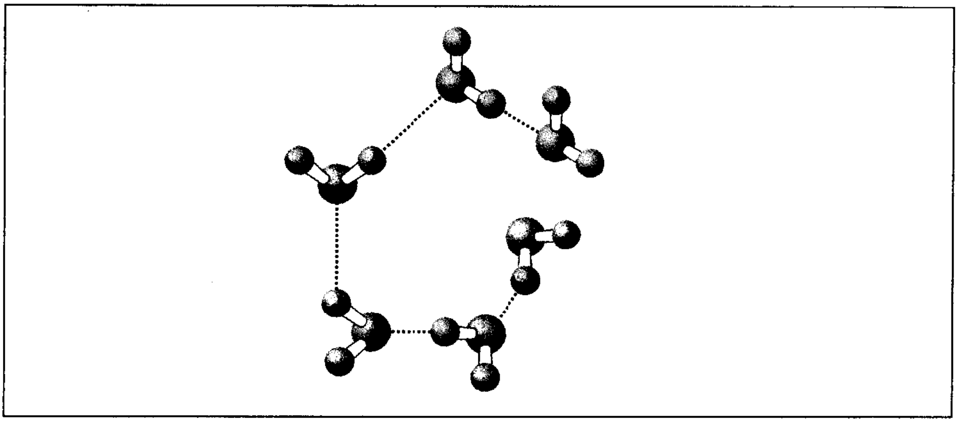


图 9-8　水中的氢键

非极性溶质越大，就需要更多的水分子重新定向来容纳它，溶解它就需要花费更多的自由能。当然，如果非极性溶质的表面有一些极性集团，水分子可以和这些集团形成氢键，而不是和其他水分子，水格子受到的破坏就少些。存在于水溶液中的球状蛋白质，尽管主要由非极性集团组成，但是它必须有一个好的可形成氢键的表面。亲水性(hydrophilic)氨基酸是那些侧链中可以为周围介质提供氢键形成集团的氨基酸，而疏水性（hydrophobic）氨基酸的侧链则没有。球状蛋白质的表面通常有50%~70%的极性原子，背离这种模式的区域可能是结合位点或络合位点。

溶剂可接近性和疏水性在评价模型结构时起着重要作用。蛋白质折叠识别的穿线方法在评价模型时使用氨基酸环境。当发现许多疏水性氨基酸处于溶剂暴露的结构环境中，或者亲水性氨基酸被包埋在蛋白质内部，则认为蛋白质模型的折叠是不正确的。

电荷－电荷、电荷－偶极和偶极－偶极的相互作用

与共价键不同，蛋白质结构中的其他重要相互作用都是非专一性的。它们并不改变相互作用原子的离散属性，也不共享电子。如果共价键结合的原子是结婚，那么非共价键的原子仅仅是同居。

几种重要的作用力是在极性和带电荷的原子间产生的。离子（ion)是因为得到或失去电子而带有负电荷或正电荷的原子。带正电荷的原子会吸引带负电荷的原子，强度的大小依赖于电荷的大小和原子间距离的倒数。在蛋白质中，电荷 – 电荷相互作用发生在碱性氨基酸侧链和酸性氨基酸侧链之间，这两种氨基酸在通常的生理条件下，因失去或得到可能的质子而带有负电荷或正电荷。在蛋白质结构中氨基酸之间的电荷 – 电荷相互作用称为盐桥（salt-bridge)。它是稳定蛋白质结构的重要作用力。

还有一些更弱的相互作用力发生在电荷与带有正电荷或负电荷的集团之间。偶极分子（dipolar molecule）是一些类似参与氢键结合的分子，分子的一端带有部分正电荷，另一端带有部分负电荷。分子的偶极是描述键的极化程度的基本载体的矢量。偶极分子可以被其他部分电荷或离子电荷强烈吸引。许多氨基酸侧链和蛋白质骨架有很强的偶极性，因此，电荷 – 偶极和偶极 – 偶极的相互作用在蛋白质结构稳定性中扮演着相当重要的角色。

范德华力

范德华(van der Waals）力是分子间的非特异性吸引力。这种力粗略看上去有点像重力，它存在于每一对非共价键原子之间，是相当长距离的作用力。然而，它不是简单地与相关原子的质量有关，而是与每一个原子瞬间偶极产生的瞬间吸引力有关。范德华力相当强，因为范德华力相互作用是非特异性的而且数量巨大，所以在蛋白质的折叠和结合中起着重要作用。

排斥力

排斥力，或者称空间(steric)相互作用，是非常短程的作用力，它随着原子中心的相互靠近而迅速增加。排斥力开始迅速增加的半径定义为环绕每个原子中心的球形边界（称为范德华半径），在边界里面另一个原子的球形边界不能通过。如果结构中两个非键合的原子相互进入对方区域。这在能量上是不利的。在真正的分子中，原子处于其他每一个原子的区域之外。然而，在分子模型中，无论是来源于NMR（核磁共振)，X射线数据，还是从头构建，检测非键合原子间的范德华碰撞都是结构提炼过程中的重要部分。

原子间作用力的相对强度

原子间的相互作用可以用偶势能(pair potential）表示，例如 Lennard-Jones 势能(见图 9-9)，包括吸引和排斥。势能的形式显示原子在很近的距离内倾向于相互排斥（正势能表示不利的相互作用），但是在稍远一些距离相互吸引。吸引强度依赖于被建模的力的类型随距离而衰减。

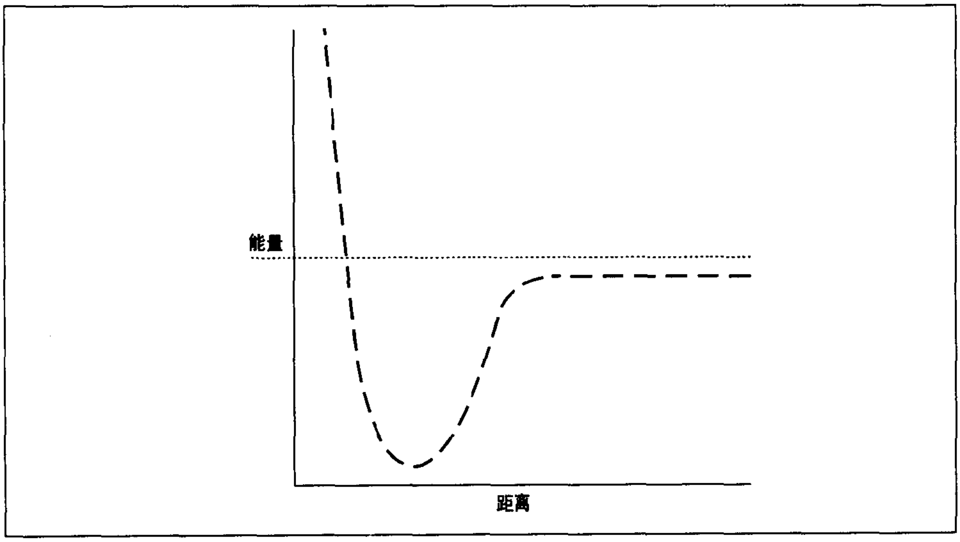


图 9-9　Lennard-Jones 势能的散点图

当根据原子间的相互作用对结构稳定性或功能进行推论时，理解这些作用的相对强度和它们怎样按距离而测量是很重要的（见表 9-1)。

表 9-1　如何根据距离测定原子间的相互作用

键的类型	相互作用的范围
共价键	复杂的短程
氢键	大约是 $1/r^2$
电荷 – 电荷	用 $1/r$ 测量
电荷 – 固定偶极	用 $1/r^2$ 测量
电荷 – 旋转偶极	用 $1/r^4$ 测量
固定偶极 – 固定偶极	用 $1/r^3$ 测量
旋转偶极 – 旋转偶极	用 $1/r^6$ 测量

表 9-1　如何根据距离测定原子间的相互作用(续)

键的类型	相互作用的范围
电荷 –非极性	用 $1/r^4$ 测量
偶极 –非极性	用 $1/r^6$ 测量
非极 性 – 非极性	用 $1/r^6$ 测 量

在表 9-1 中，*r* 代表以埃为单位的两个原子间的距离。那些强度随 *1/r* 而减弱的相互作用比那些强度随 *r* 的更高次方而减弱的相互作用力在更远的范围内起作用。共价键相互作用和氢键较强，在短距离时的能量很大。电荷 – 电荷相互作用有一些长距离作用。实验显示静电作用对蛋白质活性的影响超过 15埃，这在分子水平是一个相当远的距离。蛋白质表面的电荷密度能够产生很强的静电指导作用，在更远的距离上吸引配基分子和其他蛋白质。氢键和电荷 – 偶极相互作用也相对较强。这些相互作用的效果可以通过计算静电势能来建模，也可以通过使用计算出的势能为基础来计算其他诸如结合常数（通过 Brownian 动力学）或 pKa 值来建立模型。

另一方面，非电荷和非极性原子间的相互作用非常弱，仅在短距离上起作用。然而，这些作用是可以累加的，可以稳定结构并使分子间的结合更有利。在计算分子间接触面积的大小或枚举一个蛋白质中相邻相互作用间的相互作用时，要考虑到这些相互作用的效果。在本章的剩余部分，我们将讨论各种测量和评价蛋白质原子结构的方法，所有这些方法都可以用来增强对蛋白质化学性质的理解。

基于 Web 的蛋白质结构工具

既然已经回顾了蛋白质化学性质的基本知识，现在让我们把注意力转向工具。蛋白质结构信息的最重要来源是 PDB。除了自身作为一个蛋白质数据的登录点外， PDB 网站(*http://www.rcsb.org/pdb*) 还包括许多工具数据库链接，在搜索数据库时，可以把它们用于单独的蛋白质结构。数据库中的信息可以通过蛋白质结构浏览器界面得到。对于每一个蛋白质，可以使用像 RasMol 或 Java QuickPDB 查看器这样的三维显示工具来观看分子结构。PDB 文件和文件首部可以以 HTML形式查看，并以多种格式下载。PDB 还提供到蛋白质结构分类数据库 CATH、FSSP 和 SCOP 的链接，并提供基于结构比对的结构搜索工具 CE 和 VAST。一般的几何性质，包括二面角、

键角和键长等可以用注有极限值和偏差值的表格形式显示。序列可以按照二级结构查看和标记，序列信息可以以 FASTA 格式下载。

在 PDB 主页的浏览框内输入蛋白质的 4 字母的 PDB码，可以直接找到感兴趣蛋白质的页面。PDB 也可以用 SearchLite 和 SearchFields 两种不同的搜索工具搜索。SearchLite 是一种简单的搜索工具，它允许将一个或多个用布尔逻辑运算分开的名词术语输入到一个单一的搜索框内。 SearchFields 是一个高级搜索工具，它提供的可定制的搜索形式允许使用单独的关键字搜索每一个PDB首部字段。可以通过选择表单下部的复选框来修改或重新生成表单。 SearchFields支持在 PDB首部中的 12个最重要的字段和晶体信息中的字段。 SearchFields 也允许使用用于序列比较的 FASTA 格式、二级结构特征和短序列特征来搜索数据库。

从 Structure Explorer （结构浏览器）产生的个别蛋白质的页面， PDB 提供了通向其他工具的链接菜单。这些性质还在迅速改进中。表 9-2 提供了 PDB蛋白质页面的简要说明。如果对蛋白质结构分析工具感兴趣，建议定期访问 PDB 站点。

表 9-2　PDB 摘要信息

页面	描述
摘要（Summary）页面	本页显示PDB首部的重要信息，还有关于蛋白质组成和配体及辅助因子的化学信息。
View Structure（查看结构）	本页提供使用 VRML、RasMol 和 PDB 蛋白质浏览工具查看相关蛋白质的静态图像的链接。
Download/Display File（下载和显示文件）	本页提供几个以经典的 PDB 格式和新的 mmCIF 格式下载蛋白质结构和首部的选项。
Structural Neighbors（结构邻居）	本页链接到手工校正的蛋白质分类数据库，如SCOP和CATH，以及自动蛋白质结构比较工具 CE 和 VAST。
Geometry（几何学）	本页以表格形式提供蛋白质键长、键角和二面角的数据。
Other Souras（其他资源）	本页为每个蛋白质提供一个丰富的链接目录，从它的 SWISS-PROT 进入到描述结构的文献。从该页可以产生从结构域分析到结构质量报告，以及基因组目录和 NCBI 分类数据库的一切信息。
Sequence Details（序列详情）	本页显示蛋白质序列和从晶体数据得到的二级结构特征定位。PDB 条目中每个蛋白质的序列也可以以 FASTA 格式下载。

在下面的章节中我们将讨论链接到 PDB 站点的一些工具的特殊用法。与任何基于 Web 的工具一样，在从它们的结果中得出任何结论之前，尽可能多地学习它们的基本算法是一个好主意。因为被 PDB 认可的方法并不一定是 100% 正确的，或者是你不需要了解方法就可以解释结果的。

结构可视化

最初为结构分析开发的工具之一和你最初可能想做的分析之一仅仅是结构的可视化。蛋白质结构数据是以 x，y，z 坐标的集合的形式保存的，但是蛋白质不能仅仅通过绘制这些点而可视化。不得不考虑蛋白质中原子的连接，而且为了产生可视化的效果，需要建立能够提供深度幻觉的虚拟的三维环境。幸运的是，所有这些已经在 20 世纪 80 年代和 90 年代已完成了，现在每种操作系统都有大量免费和商品化的结构可视化工具。

尽管使用虚拟的三维表示法，蛋白质结构还是太复杂，以至于在视觉上理解它们是十分困难的。人的眼睛可以理解三维固体，但是要适应复杂的拓扑三维数据还是很困难。有许多种传统的简化蛋白质结构表示的方法允许你不用知道原子的细节，而直接看到整个蛋白质的拓扑结构。为了起到作用，蛋白质结构可视化程序需要至少能够显示用户选择的有着正确连接的原子子集，画出标准的蛋白质的卡通表示法，如条带和圆筒，根据指定的参数为分子的子集重新着色。

Web 浏览器使用的分子结构查看器

有一种分子结构查看器是一种小巧的应用程序，它可以使用 Web浏览器来工作。在进行正确的参数设置后，在 Web上访问它时，它可以显示分子数据。 RasMol和 Cn3D 是最流行的两个查看器。

RasMol

RasMol是最流行的分子结构可视化程序之一。它可以用于广泛的操作系统，以标准的 PDB 格式读取分子结构文件。RasMol 2.7.1是最新版本，可以从 Bernstein和 Sons 下载（*http://www.bernstein-plus-sons.com*）它的源代码和预编译的二进制程序。

RasMol有三种显示深度：8位、16位和32位。8位是默认值。但是如果有一台高分辨率的显示器，你可以通过实验找出哪一个最适合你的系统。但这样有一个问题，当你运行RasMol时，它会提示没有检测到合适的显示装置。使用8位的版本，可以继续你的工作。

如果计划自己编译RasMol，需要进入*src*子目录，并编辑*Makefile*文件产生合适的版本。为了这样做，可使用*vim*或者Emacs这样的ASCII文本编辑器打开*Makefile*文件，寻找变量*DEPTHDEF*。你可能发现像下面一样的东西：

```
# DEPTHDEF = -DTHIRTYTWOBIT
DEPTHDEF = -DSIXTEENBIT
# DEPTHDEF = -DEIGHTBIT
```

在这个例子中，*DEPTHDEF*定义为16位。

以#字符开头的行为注释行，当*make*程序扫描*Makefile*文件时，它不被读取。代码行可以通过注释而被跳过，也就是，通过标记为注释。除去你所要的深度定义前的#字符，把它加到其他行。注释字符在不同的编程语言中是不同的，但是注释行的概念在所有标准语言中是通用的。

根据安装向导，你也许需要编辑*rasmol.h*文件。

一旦有了合适的RasMol可执行程序，无论是下载的还是你自己编译的，都需要将它复制到*/usr/local/bin*子目录并把文件*rasmol.hlp*复制到*/usr/local/lib/rasmol*子目录。然后在Web浏览器的优先版中加入RasMol作为一项应用。如果使用Netscape（它是大多数Linux系统的默认浏览器），进入Preferences → Navigator → Applications菜单，在对话框中输入下面的值：

```
Description:Brookhaven PDB
MIMEType:chemical/x-pdb
Suffixes:.pdb
Application:/usr/local/bin/rasmol
```

也许你还想为MIME类型的文件*chemical/x-ras*创建第二个条目

当从命令行运行时，RasMol打开一个单一的具有黑色背景的图形显示窗口。这个窗口中的分子可以直接由鼠标或者用窗口下面和右面的滑动块旋转。该窗口有五个下拉式菜单。File（文件）菜单包括打开分子文件的命令。Display（显示）菜单包括

改变分子显示模式的命令，包括球形、棒形、卡通和空格填充的格式。这些显示命令执行得非常快，所以可以每一种都试一下，看看不同标准的分子显示格式。Colours （颜色）菜单允许改变整个分子的颜色方案， Options（选项)菜单可以改变显示模式，允许立体显示分子、打开或者关闭杂原子的显示或标记等等。 Export （导出）菜单允许用 GIF、PostScript和 PPM等不同的电子图像格式输出图像，随后可以用 Linux 下的许多标准图像操作程序来编辑它们，如 GIMP 等。

在 RasMol 中输入或者保存文件时，要从 RasMol 的命令行进行。在开始 RasMol 的 shell 窗口中，命令提示变为 *RasMol>* 。在这个命令提示下输入 *help* 命令，可以看到整个 RasMol 命令区，包括选择原子子集的命令。如果 RasMol 提示不能发现帮助文件，会产生一个指向 */usr/local/lib/rasmol.hlp* 的符号链接，该文件所在目录是你安装 RasMol 的目录，或者是你正在运行 RasMol的目录。帮助命令可以帮助你创建自己的颜色和显示格式的组合，包括一些菜单中没有的；创建原子间距的监视器；显示分子间的一些相互作用，如氢键和盐桥等。

Cn3D

Cn3D是来自于 NCBI的一个应用程序，它可以以 NCBI的 ASN.1 格式查看蛋白质结构文件。如果经常使用 NCBI 的数据库，你会希望安装这个工具并在浏览器中设置它为一项应用。

为了在Linux 工作站上安装 Cn3D 并把它作为浏览器的应用选项，仅仅需要从 NCBI 下载 Cn3D 压缩包，在你自己的机器上建立 *Cn3D* 目录，将文件移动到该目录，并解压缩。

然后在 Web浏览器的应用优先级选项中输入如下的新条目：

```
Description:NCBI ASN.1
MIMEType:chemical/ncbi-asn1-binary
Suffixes:.prt
Application:/usr/local/cn3d/Cn3D
```

Cn3D 会打开两个窗口：一个彩色结构查看器，在其中分子可以根据不同的性质进行旋转和着色，以不同的显示格式出现；一个序列查看器，允许查看序列，根据显示的蛋白质进行排列，为序列添加图形来突出显示二级结构的位置。

SWISS-PDB 查看器

SWISS-PDB 查看器是一个相对较新的三维结构显示和分析工具，它是由瑞典生物信息研究所提供的。它可以使用 SWISS-Model的 Web服务器为同源性建模准备输入。但是，它也可用作单纯的可视化工具。该查看器整合了许多有用的功能，包括结构的叠加、分子表面和静电位的计算、高质量透视图、扭转角的分析和结构突变的建立等等。在本书的写作过程中，SWISS-PDB正处于快速发展阶段，如果有兴趣，你可以搜索瑞典生物信息研究所的网页，寻找目前的版本和在线文档。

独立的建模软件包

重型分子结构查看器倾向于比RasMol和Cn3D这样的 Web 应用程序有更多的性质。其中最流行的是 MolMol、MidasPlus 和 VMD。这些程序可运行于台式机上。为了使用它们，需要将你感兴趣的 PDB 文件的副本储存在计算机上。

MolMol

如果你有与 Web浏览器链接的 Cn3D和 RasMol，那儿你就有及时查看任何分子结构的很好装备。可是有时需要对分子结构进行更广泛的操作。 MolMol 是一个特色齐全的分子结构可视化软件包，它允许显示分子、编辑结构和计算分子特征。

通过在命令行中输入 *molmol* 命令，可以运行 MolMol程序。没有命令行选项。该程序会打开一个白色背景的大窗口和一个独立的小窗口，包括 *x*、*y*、*z* 旋转的滑块和改变深度及剪辑平面位置的滑块。剪辑平面（clipping plane）控制显示窗口的虚拟深度和显示窗口横切分子结构的位点。原子选择选项由主窗口右侧的菜单条控制。

与 RasMol 一样，MolMol 也有下拉式菜单，但它的所有选项都可以从下拉菜单中得到，而且比 RasMol 多。MolMol 有一个完整的手册，它随着软件以 HTML 格式和几种可打印格式发布。因此我们在这里不再详细讨论每一个命令。除了标准的分子显示功能外，你还会发现其他一些有用的性质：Ramachandran 图和连接图的显示、大分子表面的计算和显示、精确的静电位的定性显示。

MolMol 可以从 ETH Zurich 以二进制发布包的形式得到，在 Linux 工作站上很容易安装。按照提供的指导进行是不会出错的。虽然 MolMol的界面不如 MIS 的 Quanta 等商业软件那样吸引人，但是它具有令人惊奇的价格。有两点一般性的提示：一定

要通过点击 OK按钮或者从菜单选择退出来关闭对话框和窗口，而不是点击窗口中右上角的关闭按钮。如果程序看起来需要花一些时间来做某些事情，不要点击许多额外的按钮或者试图强迫关闭程序，只需要等待即可。这可以保证程序不会从你的机器上挂断。

MidasPlus

MidasPlus 是一个在质量上接近商业软件的分子建模软件包，可以从旧金山的加利福尼亚大学得到。它除了提供许多标准的分子显示功能外，还提供了测量工具、有限的建模能力（如在结构中替换氨基酸的能力）和分子表面积及静电位的计算等。MidasPlus 的源代码和可执行程序可用于各种平台，包括一些 Linux 系统，只需交纳 350 美元的许可费就可从 UCSF 得到，这比相应的商品化软件包便宜得多。为了支持 MidasPlus ，你的 Linux 工作站必须配备一个高质量的三维图形卡。

VMD

另一个产生分子图形的出色的软件包是 VMD,它是以伊利诺伊大学的理论生物物理组开发的可视化分子动力学（Visual Molecular Dynamics）程序。VMD 被设计用于从分子动力学模拟观看轨迹并赋予其活力，但是它也可以产生相当好的单分子可视化效果。VMD 可用于 Linux 系统，有易于使用的菜单驱动的图形用户界面。

使用 MolScript 创建高质量的图片

用法: `molscript -in` *`infile`*`-[`*`options`*`] -out` *`outfile`*

用法: `molauto -[`*`options`*`]` *`infile`* `>` *`outfile`*

MolScript和我们已经讨论过的其他可视化软件包的目的完全不同。如图 9-10所示，它的设计目是产生用于印刷出版物的高质量图片。它被设定从命令行运行，以产生 PostScript、Raster3D和 VRML输出格式。它也可以使用 OpenGL ，以配置在自己的窗口中交互式运行，产生其他许多图像文件格式的输出（注 1）。

注 1: 图 9-10 中的图像是由 Per J. Kraulis 提供的，摘自“ MOLSCRIPT: A Program to Produce Both Detailed and Schematic Plots of Protein Structures”,《Jouranl of Applied Crystallography》(1991), 第 24期 P946~950。

在 Linux 工作站上使用 OpenGL建立交互式的 MolScript不像 RasMol 那样直接 ;它需要安装 Mesa（OpenGL 的开放源代码）文库，并定制发布包中的 *Makefile* 文件。然而，基本的 MolScript 安装还是很快捷简便的。它会产生看起来很吸引人的分子结构的卡通素描图，这是一种非常精致的适合印刷媒体的形式。为了安装 MolScript 的基本版本，仅仅按照安装文件的提示进行就可以了。复制生成的可执行文件（*molscript* 和 *molauto*）到 */usr/local/bin* 目录或者复制到默认路径中的其他目录即可。下面是 *molscript* 和 *molauto* 的功能：

molscript

MolScript主程序；产生图像

molauto

MolScript的设置程序；从 PDB 输入文件自动产生基本的 MolScript 输入文件

MolScript有两种输入文件：MolScript命令文件和 PDB 协同文件。下面是产生图 9-10 中图像的 MolScript 文件：

```
! MolScript v2.1 input file
! generated by MolAuto v1.1.1
title "MYOGLOBIN  (FERRIC IRON - METMYOGLOBIN)"
plot
  read mol "1MBN.pdb";
  transform atom * by centre position atom *;
  set segments 2;
  set planecolour hsb 0.6667 1 1
coil from 1 to 3
set planecolour hsb 0.619 1 1
helix from 3 to 18
set planecolour hsb 0.5714 1 1
coil from 18 to 20
set planecolour hsb 0.5238 1 1
helix from 20 to 35
...
coil from 94 to 100
set planecolour hsb 0.1429 1 1
helix from 100 to 118
set planecolour hsb 0.09524 1 1
coil from 118 to 125
set planecolour hsb 0.04762 1 1
helix from 125 to 148
set planecolour hsb 0 1 1
coil from 148 to 153;
set colourparts on
bonds in require residue 1 and type HEM;
end_plot
```

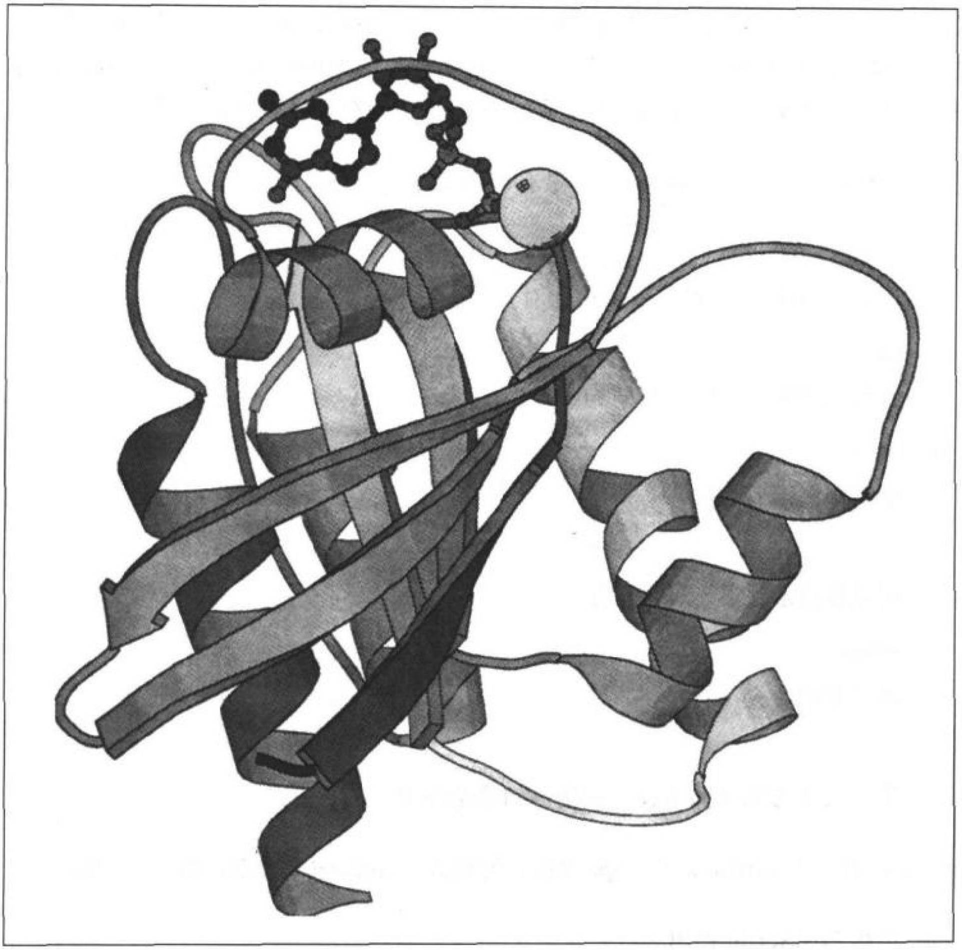


图 9-10　molscript 生成的图像例子

MolScript 脚本语言是独特的，不依赖于任何标准计算机语言。学习它的惟一方式是确定你想做什么，通过学习手册和例子来学习语言。上面显示的例子是一个简单的 MolScript 命令文件；它读取单个分子，把它放到分子团的中心，定义各种二级结构元素的位置，然后使它们的色彩按照光谱按从红色到蓝色的顺序逐渐变化。但是，MolScript 的散点图可以按比例绘制，将多个点显示在单页中。分子中原子的子集可以打开，以不同格式和定制颜色显示，也可以把标签加到图中。

幸运的是，*molauto* 程序可以自动产生 *molscript* 程序的简单输入文件，它能帮助你开始使用 MolScript 命令语言。*molauto* 可以完成输入文件中大部分乏味的工作——

为每一个二级结构片段安排螺旋（helix）、片层（sheet），或者卷曲的画图形式和颜色。*molauto* 还有各种命令行选项，可以通过输入 *molauto -h* 访问。*molauto* 以标准的 PDB 文件格式读入输入文件，除非使用重定向符，否则写到标准输出。

下面列出的是一些用于 *molauto* 的最有用的命令行选项：

-ss_pdb
　　从 PDB 文件中读取二级结构比对

-ss_hh
　　使用氢键模式安排二级结构

-cylinder
　　用筒形表示 α 螺旋

-stick
　　用球棒表示法显示辅助因子

-nocolour
　　离开着色命令

-nice
　　使用更多的颜色和片段，以提高渲染的质量

molauto 程序的输出是 *molscript* 主程序的输入。*molscript* 命令行的选项包括：

-ps　产生 PostScript 输出

-v　产生 VRML 输出

-size width height
　　改变输出图像的大小

molauto 产生的默认输入文件可以通过手工编辑产生多种效果。你可能想做的一件重要事情是旋转分子结构直到达到最好的观察效果（如果没有安装带 OpenGL 支持的 MolScript 软件包，这是不能自动进行的）。

为了使用 *molscript* 的非交互式版本来旋转分子图像，需要将当前的程序行：

```
transform atom * by centre position atom *;
```

替换成：

```
transform atom * by centre position in amino-acids
                 by rotation x    0.0
                 by rotation y    0.0
                 by rotation z    0.0
                                       ;一定要加上这个分号。
```

加入到 *molscript* 输入文件中。

在产生了图像的第一个版本以后，可使用 *gv* 这样的快速 PostScript 查看器打开它。为了改变分子的外观，可以试着改变输入文件中的 *x*、*y*、*z* 旋转值。因为 *molscript* 仅需几秒钟就可以运行任何一个蛋白质输入文件，你可以改变输入文件，保存，再显示新的输出文件。重复几次直到满意为止。

molscript 图像文件一旦产生，就可以查看，转变成其他格式，并用标准的 Unix 图像操作工具进行编辑。如果安装了大多数重要的 Linux 发布包，你可以下载一个叫 GIMP 的程序，它是一个类似于 Adobe Photoshop 的免费软件包。

使用 LIGPLOT 可视化活性位点

用法：`ligplot protein.pdb resid resid chain`

另一个为出版物产生图像的有用工具是 LIGPLOT 程序（*http://www.biochem.ulc.ac.uk/bsm/ligplot/ligplot.htm*），它可以从伦敦大学(UCL）的结构和建模小组得到。假设以一个分子结构和结构中的特殊残基或杂原子集团作为输入，LIGPLOT 会自动产生一个显示氢键、原子间的联系和溶剂可接近性的二维示意图。LIGPLOT 的例子见图 9-11。

为了在 Linux 工作站安装 LIGPLOT，仅仅按照 *README* 文件的指导进行就可以了。

为了让 LIGPLOT 正确找到它的参数文件和帮助程序，需要把一些路径信息添加到 *.cshrc* 文件中。

```
setenv ligdir /usr/local/ligplot
alias ligplot $ligdir'/ligplot.scr'
alias ligonly $ligdir'/ligonly.scr'
alias dimplot $ligdir'/dimplot.scr'
alias dimonly $ligdir'/dimonly.scr'
```

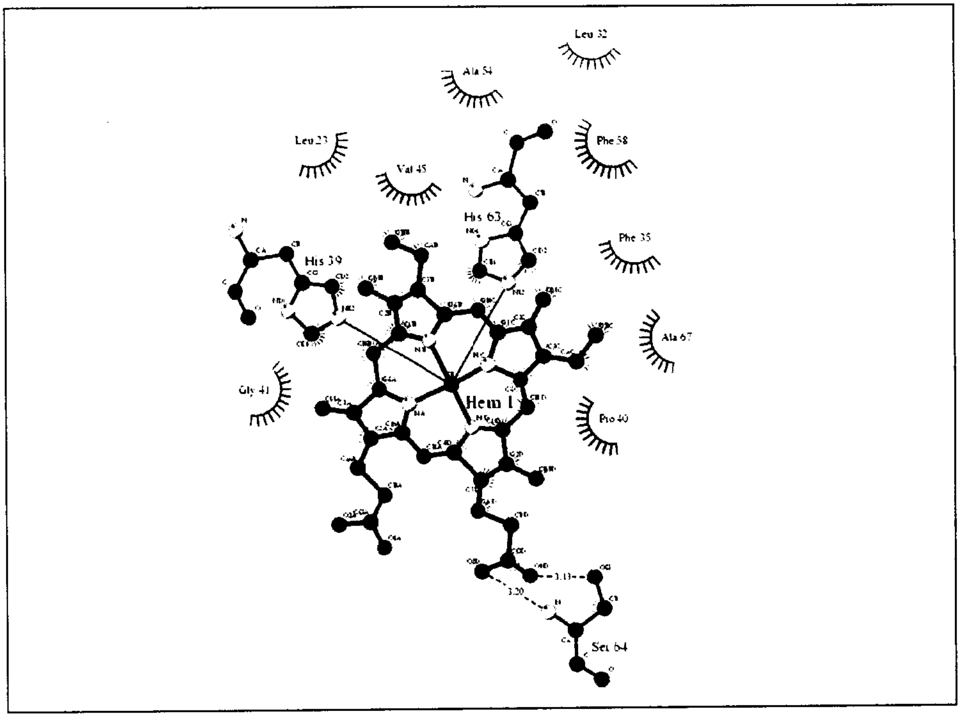


图 9-11　LIGPLOT 产生的细胞色素 B5 的亚铁血红素辅助因子配基的示意图

```
setenv hbdir /usr/local/hbplus
alias hbplus $hbdir'/hbplus'
```

命令行的值指定了特定蛋白质链中残基的范围。程序不仅可显示配基和修复团间的相互作用，也可显示蛋白质任何残基间近相互作用的网络。当选择残基的的范围很小时，它会工作得最好。

dimplot

用法: dimplot *protein.pdb chain1 chain2*

用法: dimplot *protein.pdb -d domain1 domain2*

dimplot 程序是 LIGPLOT 的一个变种，它显示两个蛋白质链或者域接口之间的相互作用，只有 PDB 文件把蛋白质标定在组织域水平上时，域变种才会起作用。

具有讽刺性的是，在 Linux 系统上安装 LIGPLOT、*hbplus* 和 *naccess* 程序使人痛苦的部分不是安装它本身，而是解开你从 UCL 下载的加密文档。文件是使用标准的 Unix *crypt* 命令加密的。这听起来很直接，但是许多 Linux 提供者在他们的发布包里没有包括 *crypt* 命令。为了在系统上使用 *crypt* 命令，实际上你可能需要安装最新版本 *glibc-2.0*。如果你不想做这些，需要在寄出你的许可协议后从作者那里请求一个 LIGPLOT *tar* 文档的解密版本。

结构分类

蛋白质的结构分类是很重要的，因为它可提供给你一个进入独立于序列相似性的蛋白质结构世界的切入点。蛋白质不是按照功能性家族分组，而是根据它们具有的二级结构的种类（α螺旋、β片层，或者两者都有）分组。在这些大类中，还可以根据蛋白质中二级结构是如何排列的分为亚类。

蛋白质分类的焦点是找到有相似化学构架的蛋白质。这与它们的序列是否是相关的没有关系。许多年来，我们从分类中知道独特的蛋白质折叠比序列家族少得多，蛋白质化学家常常感兴趣于从更广的蛋白质结构分类得到的信息，因为分析那些信息可以帮助他们更好地理解蛋白质是如何折叠的。

把蛋白质结构分为家族并不是一件很繁琐的任务。蛋白质有许多层次的结构：一级结构（primary structure），是一维序列；二级结构（secondary structure），是由蛋白质多聚体因为空间和氢键相互作用而形成的有规律的亚结构组成；三级结构（tertiary structure），是整个蛋白质的三维结构；四级结构（quaternary structure），是由多个链组成的最复杂的蛋白质结构。形成有功能的蛋白质需要四级结构。结构分类包括开发如何将二级结构单位连在一起形成域（domain）的表示方法，域是较大蛋白质结构中结构紧密的区域。把蛋白质分为域是结构分类的另一个方面。

对于如何定量地分类蛋白质的结构，还没有真正取得一致。取而代之的是，结构以诸如“希腊结构”、“螺旋束”、“α-β桶”之类的定性命名类结束。这些折叠分类在关注主要结构特征和为结构分类创建参考框架时是有用的。可是，定性分类不能令它们自动分析，这种蛋白质分类数据库仍需要专业管理者的参与。

如果仅仅关心寻找与已发布的蛋白质结构相近的结构，许多在线分类数据库中存在

的结构已经通过结合自动分析和蛋白质结构专家输入的方式进行了注释。也有一些通过序列比对寻找结构邻居的自动化程序，虽然与任何比对方法一样，在分析结果时，这些工具需要你了解比较得分的意义。

如果对自己分析蛋白质结构感兴趣，有几个结构分类的方法和工具可能会对你有所帮助。

二级结构的坐标表示

蛋白质坐标数据集不会自动地标记上 α螺旋和 β片层。蛋白质二级结构特征可以通过它们的氢键模式和骨架扭转角在合理的可信度上进行辨别。

从序列中提取二级结构的标准程序是DSSP程序。DSSP分析已知蛋白质结构中的几何学和每一个氨基酸的骨架氢键结合模式，产生一个表格式的输出，包括残基编号、序列、氢键和几何学细节。 DSSP 数据库和 DSSP可执行程序来自于 1995年的发行版，可以从EBI得到，这些可执行程序在一些老的 Linux系统上可能仍会引起与 Y2K 相关的错误。升级的 DSSP 源代码可从荷兰 Nijmengen 大学分子和生物大分子信息中心的 Gerrit Vriend处得到。

STRIDE

用法：`stride -[options] infille > outfile`

DSSP 的一个替代程序是 STRIDE。在 EMBL（European Molecular Biology Laboratory，欧洲分子生物学实验室，*http://embl-heidelberg.de/stride/stride.html/*）的Web服务器中或可下载表单形式提供。在Linux 机器上，STRIDE 的编译很容易。为程序创建一个子目录，把 *tar*压缩文件移动到子目录中，并解压缩。使用 *make* 编译程序。

STRIDE的命令行选项包括：

−M molscript file
产生一个简单的 MolScript输入

−h 报告氢键信息

-o 只报告二级结构比对

不带命令行选项直接运行 STRIDE 可以看到完整的命令列表。

STRIDE 的输出格式是结构化的 78 字符行。下面的例子说明了氢键信息的输出格式:

```
ACC  ALA -  143  142 -> TYR -  146  145  3.3  107.8  125.8  58.5  76.9  1MBN

ACC  ALA -  143  142 -> LYS -  147  146  3.2  154.3  113.4  0.1   43.4  1MBN
DNR  ALA -  144  143 -> LYS -  140  139  3.0  153.6  109.9  16.4  27.2  1MBN

ACC  ALA -  144  143 -> GLU -  148  147  3.0  160.3  109.4  11.6  6.4   1MBN
DNR  LYS -  145  144 -> ASP -  141  140  3.2  145.3  119.5  3.7   73.8  1MBN

ACC  LYS -  145  144 -> LEU -  149  148  3.0  149.4  128.8  4.7   63.7  1MBN
DNR  TYR -  146  145 -> ILE -  142  141  3.2  158.7  121.8  20.1  52.6  1MBN
DNR  TYR -  146  145 -> ALA -  143  142  3.3  107.8  125.8  58.5  76.9  1MBN

ACC  TYR -  146  145 -> GLY -  150  149  3.0  156.9  96.3   37.1  37.7  1MBN

ACC  TYR -  146  145 -> TYR -  151  150  3.1  111.2  118.0  4.2   89.9  1MBN
DNR  LYS -  147  146 -> ALA -  143  142  3.2  154.3  113.4  0.1   43.4  1MBN
```

STRIDE源代码的构建和文档都做得很好。它是一个如何分析分子几何的出色例子。每一个功能，例如，表面积计算、扭转角计算等，都存在于自己单独的程序中。如果想理解蛋白质几何性质分析的一些标准操作，我们强烈推荐学习 STRIDE 源代码。

拓扑图

拓扑图(topology cartoon)是描述蛋白质二级结构元素拓扑排列的二维表示法。拓扑图可以显示蛋白质中二级结构元素间的空间关系和连接。即使仅仅显示结构骨架或者是使用画带状图，这些关系可能在三维结构中也不容易看到。生成自己的拓扑图的软件可以在 Protein 拓扑学网页找到，*http://www.sander.embl-ebi.ac.uk/tops/*。

图 9-12中所示的拓扑图把每一个二级结构单元表示为一种形状。圆圈是螺旋，三角形是 β链。链的起始端标记为 N，末端标记为 C。每一个元素都有方向性，它可以从连接部分的绘图方式推出。如果 N端连接是二级结构元素的边缘，元素则指向画面外面。如果 N端连接是二级结构元素的中心，则指向画面的里面。

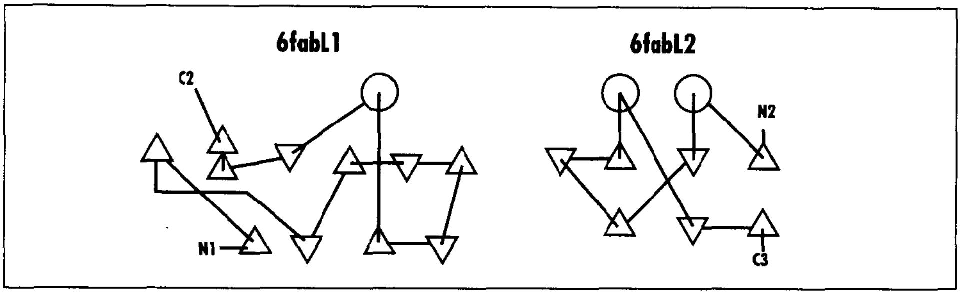


图 9-12　蛋白质拓扑图

TOPS

用法：`tops` *`pdbcode`*

TOPS 程序使用你感兴趣蛋白质生成的 DSSP 格式文件作为输入。

为了在你自己的计算机上编译 TOPS代码，需要Java语言的支持。Java 的 Linux端口可从IBM 和 Blackdown（*http://blackdown.org*　）得到。Blackdown版本需要升级到 *glibc2.1.2* ，但是 IBM 版本使用 GnoRPM 在 Red Hat 6.1 下很容易安装（当然，如果下载了 RPM 的话）。一旦 IBM 的JRE 和 JDK安装以后，TOPS的安装就没有任何困难了。EditTOPS 可执行程序允许实际查看和绘制拓扑文件，为了运行它，一定要正确设置环境变量：

PATH

　　包括 */usr/jdk118/bin*（或者是你安装 Java的任何地方）

CLASSPATH

　　安装 TOPS 类 *TOPS.jar* 的地方

TOPS HOME

　　安装 TOPS 的地方

可以通过编写一个叫做 *topssetup* 的脚本来设置环境变量，它包括如下三行，把它放在你的主目录里。在试图运行 TOPS 或者 EditTOPS 之前，使用 *Source topssetup*来正确设置环境变量。

```
setenv PATH "/usr/sbin:/sbin:/usr/jdk118/bin:${PATH:."
setenv CLASSPATH "/usr/local/Tops/classes/TOPS.jar:${CLASSPATH"
setenv TOPS_HOME "/usr/local/Tops"
```

拓扑模式也在基于Web的搜索工具中实现为数据结构，这些工具允许你比较两个结构的拓扑特征或者在蛋白质数据库中搜索相似的拓扑结构。这些服务都可以在EBI得到(*http://www.ebi.ac.uk*)。

分类数据库

分类数据库是蛋白质结构的分类，它们和早期生物学家开发的以形态学为基础的分类学很相似。根据形状和拓扑学看上去大体上一样的蛋白质比看上去有明显不同的蛋白质分到更相近的类别里。蛋白质结构类型通过视觉观察和与相似物体的比较而有一个古怪的名称（如 *Greek key beta barrel*）。分类数据库可以被看作是每一个分支点有许多分支的树，这在概念上和系统发育树很相似。

SCOP

SCOP(Structural Classification of Proteins，蛋白质结构分类，*http://scop.mrc-1mb.cam.ac.uk/scop*)是由英国哥伦比亚大学分子生物学MRC实验室维护的一个数据库。SCOP经过广泛的手工校正，内容比PDB至少落后几个月。SCOP是一个简单、技术性相对较低的资源，它由几层HTML页面组成，可以链接到个体蛋白质和折叠的图片，也可以链接到使用 RasMol或者 Chime插件打开的结构文件，还可以链接回 PDB下载结构。

在SCOP的最上层，已知蛋白质通常按照它们的二级结构特征分组为全α螺旋、全β折叠、无规则卷曲、带有金属离子的小蛋白质和各种类型的α-β混合的结构。这些主要类型被称为SCOP内的类(Class)。分类的下一层是Fold(折叠)层次的，它是拓扑结构和与已知功能域相似的域的混合：一个折叠可被叫做“球形”，另一个叫做“四螺旋的上下束”。在折叠层次的下面，蛋白质可以进一步分为超家族(Superfamily)和家族(Family)。超家族和家族的划分可能是纯功能的，也可能是因结构的不同而引起的。

CATH

CATH（*http://www.biochem.ac.uk/bsm/cath_new/*）在概念上和 SCOP 相似，但是它在 PDB分类上稍有不同。在 CATH中，蛋白质在类别（C）、结构（A）、拓扑（T）和同源性（H）超家族层次上分类。CATH 的界面是很容易驾驭的，它是一个检查各种已知蛋白质结构的出色资源。CATH 可以用 PDB 码检索，蛋白质可以在浏览器页面内显示。CATH 维护者提供了非常优秀的蛋白质结构描述的专业词汇，可使你觉得比那些有点古怪的蛋白质家族名称更有结构上的真实性。在写作本书时，CATH Web界面正在快速修改并扩充功能，以包括从可能适合 CATH 类别的未定性基因的结构比对，到分类层次的新层次的每一件事。

单一蛋白质结构数据集

PDB 充满了副本，据推测在现在 PDB 中的大约 13000个结构中，大概只有 1000个实际代表了单一的折叠。这种单一性的缺乏会使那些基于从蛋白质数据库中获取结构模式和性质的预测和分析方法出现偏差。因此，需要产生一个PDB 的非冗余子集，从一组相似的蛋白质中选择每一类别的最佳代表。这在本质上是分类问题的子问题，长期以来都是靠人工检查和注释 PDB 数据来完成的。但是随着 PDB 的增长，基于序列比较自动产生非冗余数据集的方法已经出现。

虽然各自的参数不同，但是产生这类数据集的方法相当标准。首先 PDB 通过去除极短的蛋白质链、解析度很差的链和包括大量非标准残基的链而精简。PDB 然后被分解为单独的链，链通过各种质量标准进行分类。进行全序列对全序列的比较，差别不足以达到特定临界线的链被去除，选择去除两个链中质量较低的链，直到列表中所有的链在成对比较中符合单一性标准。最后，如果被去除的链和最终集中的任何链都没有违反单一性标准，可以把它们重新加回到这个集中。

现在非冗余数据集可以从 EMBL 的 PDB Select、NCBI 和 Fox Chase 癌症中心的 Roland Dunbrack 博士处得到。据我们所知还没有软件允许我们根据自己选择的参数来创建单一的数据集，虽然上面提到的小组可能希望通过特殊要求产生数据集。用来生成序列 DB 的非冗余数据库的Perl脚本 *nrdb90.pl*，也可以从 EBI得到，然而，它很难编码产生有 90% 序列相同的水平的非冗余集。如果不怕困难，你可以为自己的目的修改这个脚本。

结构比对

最近人们做了许多努力使蛋白质结构分类成为一个自动化的定量过程，而不是一个专家校正的过程。重叠和比较结构是一个三维问题，它比比较一维结构数据需要更多的资源。然而，现存的自动结构比较工具主要是作为搜索结构比较的预先计算数据库的在线工具。

比较两个蛋白质的结构

表达两个蛋白质结构间区别的最常用参数是两个结构之间原子位置的RMSD，即均方根差（root mean squared deviation）。RMSD可作为一个蛋白质中所有原子的函数来计算，或者作为一些子集原子的函数计算，如蛋白质骨架或者只是 α 碳位置。使用蛋白质原子的子集十分常见，因为在比较两个蛋白质结构时它们的序列可能不同，因此，在位置上可以一对一进行比较的原子只有骨架原子。

这是我们讨论的第一个背景，其中分子结构的定向（orientation）变得很重要。因为蛋白质结构通常都在笛卡儿坐标内描述，它们本质上存在于一个实际的空间中，并且使用那个空间的内置方向。RMSD是一个结构中的原子和另一个结构中相同的原子间距离的函数。因此，如果一个分子起始于参考坐标系中不同的位置，则不论它们相似与否，另一个分子——两个蛋白质间的RMSD——将会很大。

为了计算有意义的RMSD，被考虑的两个结构首先必须在可能的范围内叠加。蛋白质结构的叠加通常起始于序列比较。序列比较在原子对之间建立起一对一的关系，RMSD即是从这些原子中计算出来的。为了比较结构，原子与原子间的关系实际上可能发生在氨基酸序列中没有同样的相对位置的残基之间。序列的插入和缺失可能使两个序列相互不能对齐，然而两结构的核心构架（core architecture）还是相似的。

一旦两个结构间的原子对原子关系建立起来，重合（superposition）程序的任务就是得到两个程序间的最佳重合——也就是，有可能的最小RMSD值的重合。因为蛋白质的骨架或者核心可能是相似的，但不是相同的，所以通常在比较的两个序列中不可能所有原子对都重合得很好。完全重合一对原子可能会把另外一对原子推得更远。重合算法可以优化两个结构间的方向和空间位置。

图 9-13使用 Compare3D显示了磷酸丙糖异构酶和 β-甘露聚糖酶原子结构间的最佳

比对。两个结构足够相似，可因而被分为结构邻居，它们的链轨迹比较相似，可是它们的序列只有 8.5% 相同。

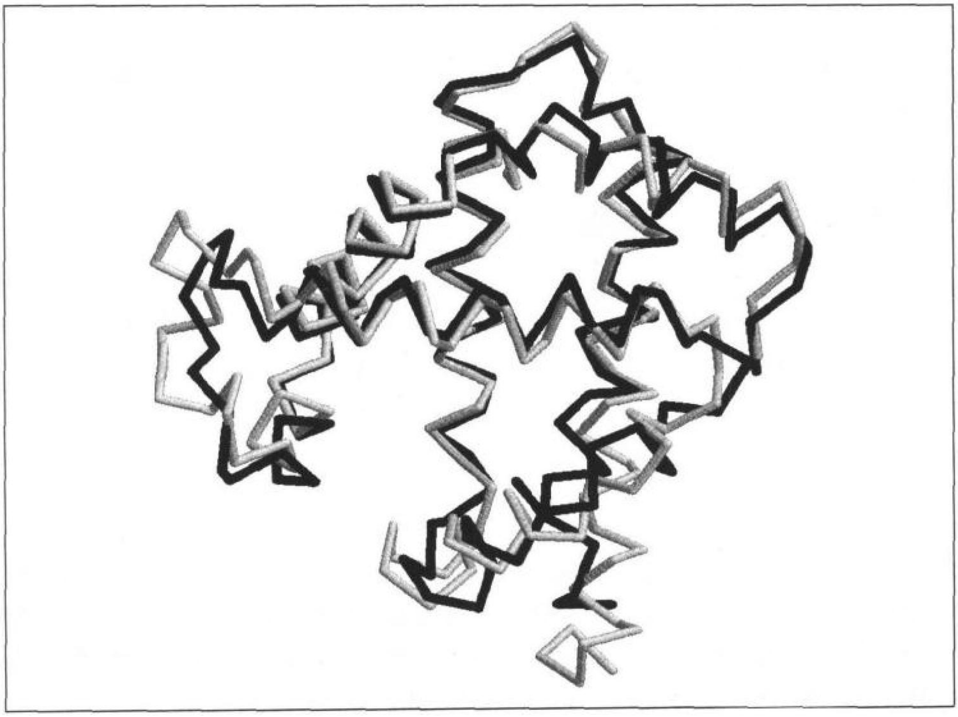


图9-13　肌红蛋白和血红素第四链的最佳重合，它们是结构邻居

一旦建立了所有结构对之间的最佳重合，计算出的 RMSD 值可以作为结果互相比较，因为在进行 RMSD 计算之前，结构已经被移动到相同的参考框架内。

ProFit

用法：`profit` *`reference.pdb mobile.pdb`*

ProFit 是英国 Reading 大学的 Andrew Martin 开发的非常易用的两个蛋白质结构阃的重合程序。一个蛋白质被使用者安排为参考结构，另一个蛋白质对参考是可变的。ProFit 输出 RMSD，也可以输出用于蛋白质重合的坐标。ProFit 允许重合选项只选择每个蛋白质的区域以使得域可以独立检查。ProFit 可编译和运行在任何 Unix 工作站上。ProFit 可以从 Andrew 的网站下载（*http://www.bioinf.org.uk*）。

DDD

EBI的 DDD（DALI Domain Dictionary，DALI 域词典）是以通过序列标识为蛋白质域自动分类为基础的。与使用人工设计的分类方案不同， DDD是在抽象的折叠空间中通过聚类蛋白质邻居而构建的。 DDD没有针对整个蛋白质，而是对紧密的可重现结构（称为域，*domain*）进行分类，这些结构可能会在不同的蛋白质内，或蛋白质之间重复出现。DDD的内容对你来说可能和 FSSP一样熟悉，FSSP是基于蛋白质结构间比对的折叠分类数据库。

DDD可用文本关键字来搜索；也可以作为树或者可点击的折叠空间的图形表示来查看。可以通过 DDD 界面查看保守域的序列数据及与其结构邻居的关系。

在 DALI/FSSP 中产生结构比对的重合程序（ SUPPOS）可以在蛋白质结构分析工具软件包 WHAT IF 中得到，我们将在后面的“ WHAT IF/WHAT CHECK ”小节中讨论该程序。

CE 和 CL

最佳途径的 CE（Combinatorial Extension，组合延伸）是一种复杂的自动结构比对算法，它使用“ 种子 ”结构比对的局部几何特征，然后把这些局部相似的区域结合成整个比对的最佳途径。动态规划可以优化比对。

CE 可以作为 Web服务器或者源代码从圣地亚哥超级计算机中心得到。 Web 服务器允许上载相互间的成对比较或和 PDB中蛋白质的成对比较的文件，来比较一个结构和PDB中所有的结构，将一个结构与代表链列表进行比较并审查特殊蛋白质家族的比对。CE也是完全和PDB 站点整合在一起的，CE搜索可以直接从序列搜索得到的任何蛋白质产生的网页开始。你可以和源代码一起下载一个包含所有PDB结构的当前的、预先计算的成对比较数据库。然而，如果仅想做几个比较，你可能不愿意这样做。

使用 CE服务器计算相似性时，可以设置几个参数，包括序列相同百分数、有空位的比对的百分数和两条链间长度不同的百分数的截止点。也可以设置 RMSD 和 Z-分数的截止点。Z-分数用于测量一个比对相对于随机比对的显著性，和 BLAST 的 E- 值类似。大于等于 3.5的 CE的 Z-得分通常说明两个蛋白质有着相似的折叠。

除 CE 外，SDSC 还提供了 CL（Compund Likeness，复合相似性）服务器，该 CL 是一套用于蛋白质结构之间概率比较的工具。在 CL 中，可选择整个蛋白质结构或者结构片段作为探针搜索 PDB，搜索的特征包括键长和键角参数、表面极性和可接近性、二面角、二级结构、形状和预测的 α 螺旋及 β 片层的系数。CL 允许你问问题“还有和我感兴趣的蛋白质（或片段）有化学相似性的其他蛋白质（或片段）吗？”，并允许定义非常广泛的化学相似性。关于 CL 的完整教程可以在 CL 的网站上得到（*http://cl.sdsc.edu/cl1.html*）。

VAST

VAST 是 NCBI 提供的成对结构比对工具。对于结构比对的参数，VAST 报告的和 CE 略有不同，在一些重要方面使用的算法也不同。可是，结果却颇为相似。VAST 搜索自动允许你在 Cn3D 浏览器插件中查看你的重合的蛋白质结构，使用 Cn3D 中显示的比对序列。在实际应用中，CE 或 VAST 都足以使你了解两个结构是如何相匹配的；如果你关心算法的不同，两个小组都在他们的网站上提供了进入详细解释的入口。与 CE 不同，VAST 软件不能下载，因此如果你想在自己的服务器上进行大量比较，CE 可能会更好。

结构分析

蛋白质结构的几何分析有两个主要目的。一是验证蛋白质结构的化学正确性，另一个是决定结构模型是否要提交给 PDB 用于分析已存在的结构。几何分析也允许你检查蛋白质结构的内部联系，因为蛋白质功能经常依赖于蛋白质结构中不相邻的氨基酸间的相互作用。联系分析可以提供对蛋白质中复杂、非序列结构模式的理解。

分析结构质量

几何分析可以显示从 X 射线晶体学数据和核磁共振数据得到的模型在哪里偏离了化学原理。像前面提到的，物理原理控制着分子间的相互作用：非键合的原子相互只能离得这么远，因为随着两个原子被迫靠近越过范德华半径时，相互的能量变得非常不利。这些相互作用限制的不仅是蛋白质链中不同部分的原子对间的联系，也限制了自由原子如何围绕连接它们的键转动。原子轨道结构和原子间键的属性限制了

键合原子间的相互位置。因此，实际上键角和二面角被限制到一组有限的值中。几何分析工具一般由晶体学家开发来显示他们的结构模型在哪里偏离了这些自然原理；它们也被同源性建模者和从头开始结构建模者用来评价结构模型的质量。

有多种工具用于分析结构质量。有些单独运行；其他一些结合成更综合的结构分析和模拟软件包。在 PDB 网站上可以找到这些工具中最好工具的详尽列表。

PROCHECK

PROCHECK(*http://www.biochem.ucl.ac.uk/~roman/procheck/procheck.html*)是一个检查蛋白质质量的流行的软件包。它可生成容易解释的彩色 PostScript 散点图来描述蛋白质结构，也能比较两个相关的蛋白质结构。该软件运行于 Unix 系统，也可以移植到 Windows 中。

使用 PROCHECK 需要在 *.cshrc* 文件中设置几个别名。需要的别名如下:

```
setenv prodir /usr/local/procheck
alias procheck'$prodir /procheck.scr'
alias procheck_comp'$prodir /procheck_comp.scr'
alias procheck_nmr'$prodir /procheck_nmr.scr'
alias proplot'$prodir /proplot.scr'
alias proplot_comp'$prodir /proplot_comp.scr'
alias proplot_nmr'$prodir /proplot_nmr.scr'
alias aquapro'$prodir /aquapro.scr'
alias gfac2pdb'$prodir /gfac2pdb.scr'
alias viol2pdb'$prodir /viol2pdb.scr'
```

各种 PROCHECK 命令脚本都需要别名，因此不能仅仅通过键入每一个单独模块的全路径名来运行 PROCHECK。在运行 PROCHECK 或 PROCOMP 时，实际运行的程序是一个命令脚本，它会调用其他几个程序和脚本。

通过编辑*procheck.prm*文件，PROCHECK可以设置为产生几种不同类型的输出（彩色或黑白)，*procheck.prm* 文件位于你将要发布 PROCHECK 命令的目录中。参数通过把 Y 变成 N 而被编辑，反之亦然。在*procheck.prm*文件中可以找到这些参数选项。文件是自我解释、易于理解的。作为参考意见，文件最重要的部分可能是你开关各种类型的散点图的部分。

procheck.prm 中的其他参数主要是用于不同类型的散点图的默认颜色值。

```
Colour all plots?
-----------------
Y <- Produce all plots in colour (Y/N)?

Which plots to produce
----------------------
Y    <-  1. Ramachandran plot (Y/N)?
N    <-  2. Gly & Pro Ramachandran plots (Y/N)?
N    <-  3. Chi1-Chi2 plots (Y/N)?
N    <-  4. Main-chain parameters (Y/N)?
N    <-  5. Side-chain parameters (Y/N)?
N    <-  6. Residue properties (Y/N)?
N    <-  7. Main-chain bond length distributions (Y/N)?
N    <-  8. Main-chain bond angle distributions (Y/N)?
N    <-  9. RMS distances from planarity (Y/N)?
N    <- 10. Distorted geometry plots (Y/N)?
```

一旦编辑 *procheck.prm* 文件至满意后，用命令 *procheck filename.pdb [chain] resolution* 运行 PROCHCK。分辨率参数在 PROCHECK 输出中使你的蛋白质和 X埃分辨率的参考蛋白质进行比较。这个参数不是任选的。 PROCOMP 的命令行需要第二个蛋白质的文件名和链 ID，而不是分辨率参数。

WHAT IF/WHAT CHECK

WHAT IF 是由 Gert Vriend 开发的多功能菜单驱动分子建模软件包，现在可从 Nijmegen 大学得到。WHAT IF 可以计算本章我们讨论过的任何蛋白质性质，从溶剂可接近表面积、pKa值、接触(contact)，直到使用 GROMOS 的分子动力学。对学术派用户来说，完整的 WHAT IF 软件包只需付很少的费用，它可以在 Linux 系统上很好地编译和运行。

WHAT CHECK 提供了进入 WHAT IF 结构质量检查子集的入口。WHAT CHECK 报告立体化学性质、键长、键角、二面角和其他质量。对 PDB 中任何蛋白质的完整 WHAT CHECK 报告都可以从 EMBL 的 PDBREPORTS 数据库下载。对于还没有储存到 PDB 中的模型和结构来说，WHAT CHECK 也可以作为 EMBL 的 Biotech Validation Suite Web 服务器的一部分得到。

分子内的相互作用

几何分析对于理解蛋白质的折叠和功能十分有用。在这种情况下，感兴趣的几何学不是蛋白质链中相互邻近的原子间的化学键作用，而是在蛋白质链中广泛分离的非

键合原子间的相互作用。结构域中结构核心的分子内接触的密度可能非常不同于两个结构域间的区域的接触密度。测量整个蛋白质中的这种密度会给我们提供蛋白质折叠过程的线索。把蛋白质结合在一起的氢键模式可作为蛋白质折叠的鉴别签名。在蛋白质特定的重要化学残基间的接触可能会提供关于蛋白质催化机制或功能的假设。蛋白质工程师可能希望检查蛋白质中的分子内接触来确定最不可能破坏蛋白质结构的变化的位置。

使用 HBPLUS 计算接触

分子内非键合相互作用和氢键相互作用列表可以使用独立的程序 HBPLUS计算，该程序可从 UCL的分子生物学结构小组得到。得到HBPLUS 程序并运行它很简单，但是由于产生的结果是一个长文本文件，它们需要经过脚本的后加工后才能使用。Weizmann 研究所（在以色列的 Rehovot，*http://pdb.weizmann.ac.il:8500/oca-bin/ipcsu/*）的 LPC-CSU（配基－蛋白质接触/结构单元接触）服务器可以产生任何蛋白质的重要分子内和分子间相互作用的文本报告。蛋白质结构也可以从用户机器上载到服务器或者使用它们的 PDB ID 码在服务器上找到。

接触作图和显示功能在 WHAT IF 软件包中也可以找到。二维接触图是大多数分子建模软件包的标准特征。二维接触图是简单的残基间成对相互作用的散点图。在图中，蛋白质残基数画在每一个轴上，点（也许用编码颜色来表示接触距离）画在 X 残基和 Y 残基相接近的任何地方。接触图有明显的模式可帮助鉴别蛋白质折叠，根据它们的序列和预测的二级结构特征，人们试图努力预测未知结构蛋白质的接触图。

溶剂可接近性和相互作用

溶剂可接近性表面的计算可帮助你指出哪些化学集团位于蛋白质的表面。蛋白质表面的氨基酸通常是决定它如何与其他分子（如化学底物、配基、其他蛋白质和受体）相互作用的氨基酸。如果知道蛋白质的化学表面看上去像什么，你就能够使用这些信息来帮助确定为什么一个分子会与另一个分子结合，为什么酶对特定底物是专一的，或者蛋白质是怎样用其他方式影响环境的。

分析形状的计算也会帮助你描述蛋白质表面的几何性质。许多生物化学教科书用锁和钥匙来描述分子间的相互作用。分子以几何专一性的方式相配在一起，因此锁的

形状（如酶）和钥匙的形状（如底物）是互补的。细胞表面的受体的形状必须和对应的配体的形状互补，否则细胞反应就不能启动。免疫系统是个很好的例子。在免疫反应中，生物产生攻击特定形式抗原的抗体，这就是为什么可以通过注射灭活病毒样品使动物产生对某种疾病的免疫力的原因。灭活病毒的形状和活的致命的病毒相似，但是它不会伤害动物。无论如何，动物产生了识别灭活病毒形状的抗体。那么，当活的病毒来到并要入侵时，动物已经有了正确形状的抗体来攻击活病毒。

因此，如果你想设计新的疫苗或者产生执行特殊反应的蛋白质，或者理解代谢途径中的两个蛋白质是如何相互作用的，能够测量分子的形状就很重要。

计算溶剂可接近性的标准方法十分简单，分子结构中的每一个原子用球表示，每一特殊原子类型的球的半径都不同。围绕已知原子中心的球通过收集数百个离散点而被建模。为了确定蛋白质溶剂可接近性表面，溶剂可接近性计算模拟半径相当于水分子半径（1.4Å）的球“探针”滚过原子球表面。探针中心的路径决定了分子溶剂可接近的表面，因为探针（此处为水分子）不能插进分子表面的较小缝隙里，计算出的溶剂可接近性表面比实际的分子表面更光滑（见图 9-14）。

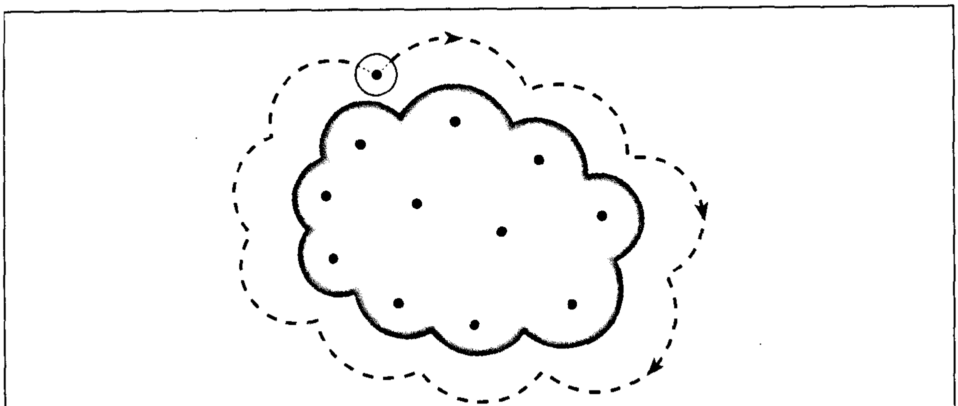


图 9-14　通过探针滚动确定溶剂可接近性

因为蛋白质是动力学实体而不是溶剂可接近性计算时假设的钢体，所以分子内部比探针滚动算法计算的可能更容易与溶剂接触。然而，溶剂可接近性计算有助于理解蛋白质分子，从而提示下一步实验。可接近性计算是得到蛋白质复杂的物理化学性质的一种方法；蛋白质的表面性质会影响它与周围介质和其他蛋白质或底物的相互作用。

用 naccess 计算溶剂可接近性

用法: naccess *pdb file* [-p *probe size*] [-r *vdw file*] [-s *stdfile*] -[*hwyc*]

有许多程序可使用探针滚动法计算溶剂可接近性。它们都很直接并易于使用，需要一个标准的 PDB文件输入，通常会以蛋白质中每个氨基酸或原子的可接近性百分比的形式作为输出。一个流行的程序是 *naccess*，它可以从 UCL的生物分子结构和建模小组得到。*naccess*可以和该小组开发的其他程序结合使用，如 HBPLUS（计算分子间相互作用和氢键的一个程序）和我们以前讲过的LIGPLOT。它也可以单独运行。*naccess*是用 FORTRAN语言写成的，因此你需要在机器上安装 *g77* 编译器来编译它。

*naccess*输出两个文件，*.asa*文件包括分子中每个原子的可接近表面积（accessible surface area）；*.rsa* 文件包括每个氨基酸的可接近表面积和相对可接近性（relative accessibility）。该程序可以处理蛋白质和核酸分子，产生单独的分子链或者整个结构的可接近性报告。*-h*、*-w* 和 *-y* 标志可以使程序分别忽略氢原子、水分子或杂原子。若带选项 *-c* 运行，*naccess* 会产生分子间的接触面积而不是可接近面积。

SURFNET 是 UCL 的 Roman Laskowski 开发的程序，它可以处理溶剂可接近性信息，显示表面特征、裂缝、空穴和结合位点的有用表示法。SURFNET 产生的输出格式可以用 RasMol 等分子可视化程序显示。

使用 Alpha Shapes 软件计算溶剂可接近性

Alpha Shapes软件是计算溶剂可接近性的标准的Connolly 表面方法在数学上准确的替代者。Alpha Shapes 是由 NCSA（*http://www.alphashapes.org/alpha/*）的数学家 Herbert Edelsbrunner研究小组开发的，用于物体表面建模的程序。Alpha Shapes 的一套专门分析蛋白质分子的扩展程序也是可以得到的。

Alpha Shapes 方法根据围绕描述结构的点的集合的空间的精确几何分解，构建一个称为结构的单纯复形或 α 复形。一旦蛋白质结构的 α 复形被建立起来，包含和排除的算法可以准确地描述结构的表面积或体积，还有空穴、裂缝和接触区。使用 Alpha

Shapes算法计算蛋白质形状的主要优点是该软件和一个复杂的可视化程序*alvis*结合在一起，后者可以显示诸如离子通道的内部形状或者蛋白质内部空穴等几何性质。

Alpha Shapes发布包由几个程序组成。这些程序必须按照合适的顺序运行才能正确地分析分子数据:

pdb2alf

将 PDB 文件翻译成 α 数据文件。

delcx

根据 *pdb2alf*输出计算分子的 Delaunay 复形。

mkalf

从 *delcx*计算出的 Delaunay 复形计算 α 复形。

VOLBL

使用 *mkalf*计算出的 α 复形和原始 PDB 文件的信息计算蛋白质性质。依靠使用的命令行选项，VOLBL可以计算蛋白质内部的空穴和蛋白质的空间填充模型，以及分子和空穴的体积。多个 VOLBL 运行可能产生互补的数据集，它们可以被加减以决定接触面积和其他分子性质。

可以在伴随Alpha Shapes 发布包的 README 文件中找到这些程序中每一个的用法细节，或者试着不附加命令行选项运行程序。

使用 *alvis* 可视化 Alpha Shapes 数据可能相当有趣。为了这样做，你需要来自 *delcx* 和 *mkalf* 的输出，而不是来自 VOLBL 的输出。为了根据来自 *molecule.PDB* （包括输出文件 *molecule.dt* 和 *molecule.alf* ）产生的数据集来运行 *alvis*，可输入 *alvis molecule*。可视化程序打开显示分子的突起的外壳。从当前版本的 *alvis* 中不能看到分子的标准原子结构，但是可以比较 *alvis* 图像和相同分子的其他图像（也许是使用RasMol 或类似的分子可视化程序）。

在 *alvis* 控制面板的左下角，可以看到一个框，该框包含一个有三种颜色曲线的图谱。该图叫做 α 等级（rank） 图，它可用来选择想查看的分子图像。把光标放到这些图像的峰、谷，或者交叉点，可看到你最感兴趣的分子形状的图像。

使用来自于 *alvis* 的 Gizmos 下拉菜单中的 Pocket Panel，可以对蛋白质中的空隙

(void)、凹穴(pocket)和不同区域的显示进行选择。*http://www.alplhashapes.org* 处的在线 *alvis* 指导手册完整地描述了查看这些特征所需的设置。

除 Alpha Shapes 的主程序外，还提供了许多实用脚本，可用来后处理 *VOLBL* 输出以得到特殊的信息。这些包括：

aacount
: 从一个 VOLBL 输出文件计算一个残基的面积或体积。

aadiff
: 计算两个模型在可接近性面积方面按残基比较的区别。

aanonpolar
: 输出非极性原子的面积或体积；*aapolar* 对极性基团做同样的事情。

areadiff
: 计算两个文件间在面积方面按原子比较的区别。

还包括计算体积差别的类似脚本。

基于 Alpha Shapes 的分析性表面势能也可以从芝加哥的伊利诺伊大学的 CAST-P Web 服务器访问。在写作本书时，并不是 PDB 中的所有蛋白质结构都出现在 CAST-P 上；这个站点目前正处于发展之中。然而，预计在将来它会成为有用的分析工具。CAST-P 的特色是一个描绘蛋白质结构中空穴和与空穴接触的氨基酸的基于 Java 的集成可视化环境。

计算物理化学性质

我们已经讨论了蛋白质分子中控制单个原子间的相互作用的力。然而，为了理解分子间的相互作用，学会理解蛋白质中的所有原子如何在一定距离上共同起作用来影响其他蛋白质或配基可能会更令人感兴趣。

物体的静电势能是该物体对附近的其他物体所产生的力的测量。蛋白质分子的静电势能是长程作用力，它可以在高达 15埃的范围内对环境中的其他分子的行为产生影响。

大分子内的静电相互作用也很重要。蛋白质内附近的带电基团可能引起氨基酸 pKa 值（酸性基团或者碱性基团失去或者得到一个质子的 pH 值点）的变化，产生该分子行使化学功能所必需的化学性质。

大分子静电学

蛋白质分子可以认为是非导电介质中的电荷的集合。在计算蛋白质分子静电势能的模型中，每一个原子代表一个带部分原子电荷的点。蛋白质溶剂可接近性表面形成了蛋白质内部介质和围绕蛋白质的外部介质之间的边界。

计算一个蛋白质结构的静电势能允许你去预测诸如单个氨基酸的 pKa值、溶解能和大概的分子间结合能之类的值。如果你对蛋白质建模感兴趣，大分子的静电势能可能是你想进一步探索的课题。我们在《Methods》杂志 2000 年 3 月的该题目的综述提供了进入分子静电势能文献的切入点。

Houston Brownian Dynamics 大学（UHBD）的软件包是一个计算大分子静电势能的艺术级的软件包。UHBD 计算静电势能，并可以使用这些势能作为参数用于下面的 Brownian动力学和分子动力学模拟。 UHBD 最近的发布包可以在 Unix 系统上编译，包括几个装备 UHBD的控制原本用于计算蛋白质中滴定单个氨基酸的 pKa 值和整个蛋白质的理论滴定曲线。UHBD 可以通过脚本语言访问；它需要一个蛋白质文件和命令脚本作为输入。它也需要一个包含任何氨基酸的原子部分电荷和输入结构中其他原子的文件。详细的脚本例子由 UHBD发布包提供，同时提供的还有电荷数据文件，它允许程序将正确的部分原子电荷赋给所有但不平凡的原子集团。

UHBD，和其他类似程序，如 DelPhi（它只和 UHBD的静电功能有重叠）使用数量近似来解决组成蛋白质的大量电荷相互作用的 Poisson-Boltzmann 方程。在 UHBD 使用的有限不同的方法中，蛋白质分子中无规律间隔的电荷画进了有规律的三维栅格中。Poisson-Boltzmann方程对栅格中的每一点反复求解，直到结果收敛到每一点的静电势能中。

使用绘制的性质进行分子表面的可视化

alvis 不能真实显示分子表面，而是显示数学化的假表面，除 *alvis* 外，显示分子表面还有其他几种选择。大多数分子建模软件包整合有显示分子表面的功能，允许根

据化学性质对表面着色。可是不是专门设计用于该目的的程序的显示方案太简单，不能处理大分子静电势能计算和其他物理化学性质表示法的数据。一个例外可能是SWISS-PDBViewer(前面讨论过)，它可以解释来自外部静电势能计算和分析分子表面计算的数据。

GRASP/GRASS

GRASP是由哥伦比亚大学 Barry Honig 小组开发的高质量的分子表面可视化程序。GRASP 可以读取静电势能的文件并把它显示为分子表面特征。它还有许多显示选项用于产生真正漂亮的静电性质的可视化解释。不幸的是，GRASP 只能用于SGI IRIX工作站，并且目前也没有计划使它用于其他操作系统。

如果使用苹果机或者 PC机，GRASP 的一些功能可以通过哥伦比亚大学的 GRASP Web界面访问。然而，这个 Web界面严重地依赖于GRASP 本身(在 SGI工作站上)，或者是 Chime 浏览器插件，或者是 VRML Viewer的界面，如果你工作在 Linux 系统上，所有这些都还有问题或者根本不存在。我们曾经成功地使用 *vrmlview*观察器和 Netscape 观看了来自 GRASS 服务器的 VRML 模型，虽然图像的质量相对较差。为了使用 *vrmlviewer*，可下载并安装程序，然后设置 Netscape的 preferences 项，使用 *vrmlviewer*可执行文件处理 MIME 类型的 *model/vrml*文件。Netscape 应用程序列表中的“Handled By”条目应该读取 */usr/local/bin/vrmlview %s* （或者是安装 *vrmlview* 的任何地方)。

GRASS 界面很简单并且是可点击的。你可以从几种分子显示选项中选择：CPK 表面、分子表面、带状或棒状模型。然后该性质可以选择绘入分子图像。可得到的性质包括使用 GRASP 内置的 FDPB 解析器计算出的静电势能、表面曲率和蛋白质序列家族中的氨基酸可变性。 GRASS未实现 GRASP 的完整功能，但是具有许多最有用的性质。

结构优化

蛋白质结构的优化是使结构和一些几何参数的理想设置相一致的方法。与前面我们在讨论结构质量检查时提到的一样，蛋白质结构模型有时会违背化学定律。将原子放置得太近会导致不适合的分子间相互接触或者范德华力碰撞。蛋白质中原子的键

长、键角和二面角也可能有错；也就是说，它们可能在期望的那类键或者角的通常的范围值之外。

结构优化对于理论模型的开发者和用实验确定蛋白质结构的研究者来说都是一个重要问题。从某种重要意义上讲，所有蛋白质原子的坐标就是结构模型。结构优化工具很久以来就是 X 射线晶体学家工具箱的一部分。结构优化的过程可能需要大量的运算。因为蛋白质结构中的所有原子都通过刚性固定长度的键相连接。移动蛋白质结构中的一部分原子会对邻近的原子产生广泛的影响。常常是将蛋白质的一部分移动到更好的构象就意味着这个蛋白质的其他部分进入一种不合适的构象。优化从本质上讲是通过一系列重复的微小改变最终汇聚成最好的整体结果。有许多优化方法，在理论计算机科学中它们都有自己的分支学科。

不需要总是知道优化方法的细节，但是如果开始频繁使用结构优化和分子模拟方法，应该意识到你对优化算法的选择可能会是个问题。优化并不是总能提供给你更好的结构模型；如果方法是基于不正确的结构规则，或者规则被不正确地区分先后次序，实际上优化可能会给你一个比你开始的模型还要错误的模型。

信息学在优化中的角色

优化中使用的理想参数或者限制是什么呢？在一些例子中，它们完全基于化学原理：通过空间限制决定的键长和键角，与 Lennard-Jones势能描述的非键合相互作用。在其他一些例子中，结构限制基于从已知蛋白质结构数据库中得来的信息。如果在一个特定序列上下文关系中的特定氨基酸总是采用同样的构象，将它预测为那种构象的概率更高，而不是不同的构象。二级结构预测方法使用基于信息的方法来预测蛋白质骨架可能的构象。一旦骨架轨迹被确定，优化方法使用信息在单独侧链的水平上提炼原子结构。

旋转异构体库

旋转异构体库是专门用于分子模型建立中侧链位置优化的参数组。它们被叫做旋转异构体（rotamer）库是因为它们包含允许远端氨基酸侧链原子围绕 Cα－Cβ 键转动的信息，表示为侧面二面角的允许值。

因为键旋转的空间限制，蛋白质中氨基酸侧链可以推测只有少数构象没有能量不适合的结果。旋转异构体库可以从化学键和角的限制派生出来，但是，它们更可能通过分析已知蛋白质结构中氨基酸侧链的构象而被开发。旋转异构体库可以是骨架依赖性的也可以是非依赖性的。骨架非依赖性旋转异构体库把所有特殊氨基酸的例子分为相同组的一部分，甚至是一个出现在β片层，另一个出现在α螺旋中也是如此。另一方面，骨架依赖性旋转异构体库根据氨基酸出现在不同的二级结构对它们进一步进行分类（注 2）。

SCRWL ，可以从 UCSF 的 Fred Cohen研究小组得到，它是一个允许使用骨架依赖性旋转异构体库建立侧链构象模型的程序。

PDF

概率密度函数的起源在概念上和开发旋转异构体库相似，虽然它在数学上更精确一些。PDF的本质是开发用于代表离散值分布的数学函数。组成分布的离散值来自于在样品代表数据库中存在的位置。这种数学函数然后可用于评价和优化（有时甚至是预测）同样位置将来出现的性质。

在蛋白质建模中，PDF 已经用于描述残基间和残基内的原子间的距离和键角、二面角和蛋白质结构的其他空间延伸区域。第十章将讨论的建模器使用键角和二面角 PDF 的结合来优化它自己建立的蛋白质结构模型，建模器的内部 OPTIMIZE例程可以用于基于 PDF 的结构优化。

产生 PDF 的数据可以分解成特殊的存在；例如，当两个残基都是亮氨酸，残基 *i* 的 Cβ 和残基 *i+4* 的 Cβ 间会完全接触，但仍会在分类细节和类别总数间权衡。蛋白质的距离 PDF 已经被几个小组用于评价和优化蛋白质结构。大多数这种工作还处于早期阶段，软件还不能让公共用户使用。

图 9-15显示了一个从已知蛋白质结构产生的半胱氨酸残基硫原子间的三级相互作用

注 2: 当结构评价和优化原则来自于数据库中已经存在的模式时，就会出现在已存在事物的高度专一性分类和每种类型的数据集大小之间权衡的情况。数据集的数据越多，规则的值可能会越好；然而，存在事物的分类专一性越差，则可能有越少的值用于预测。这不仅对旋转异构体库规则是适用的，对于 PDF和其他数据库来源的规则也是如此。

的距离概率密度函数的散点图。靠近 2 埃的函数峰对应着半胱氨酸残基的硫原子间可以形成二硫键的高概率。这些数据来自于圣地亚哥超级计算机中心的生物学工作台（*http://workbench.sdsc.edu*）和使用 *xmgr* 的散点作图。应注意的是工作台的 PDF 形成了参与二硫桥的半胱氨酸残基（这里绘制和指出的是工作台站点的 CSS 残基）和没有参与二硫桥形成的残基间（工作台站点称之为 CYS）的区别。

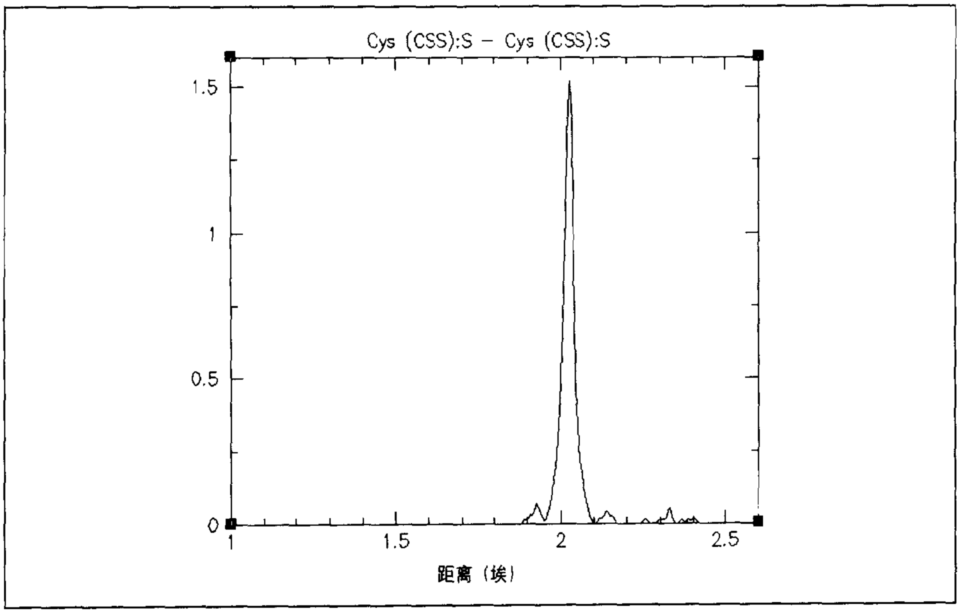


图 9-15　原子间距离的概率密度函数

基于 PDF的结构评价在 SDSC 生物学工作台的 Structure Tools （结构工具）部分已经实现（见图 9-16）。你可以上载一个 PDB结构或者理论模型，通过残基 – 残基或者原子 – 原子间的偏差为结构打分。得分可以显示为散点图，其中 Y轴代表被评价结构区域的相对概率。这可以被认为根据概率，如果已知在相似序列环境中残基或原子的其他存在情况，特定残基或原子处于“正确”位置。低概率区可能是错误的折叠或者不好的建模。PDF 概率得分也可以输出到特殊的PDB 文件中，代替原始文件中的温度因子值。然后，这些特殊的 PDB 文件可以使用 RasMol 或者 Chime 这样的可视化程序显示。用温度因子着色分子可以把 PDB 概率得分绘制到分子结构中，突出得分差的结构区域。

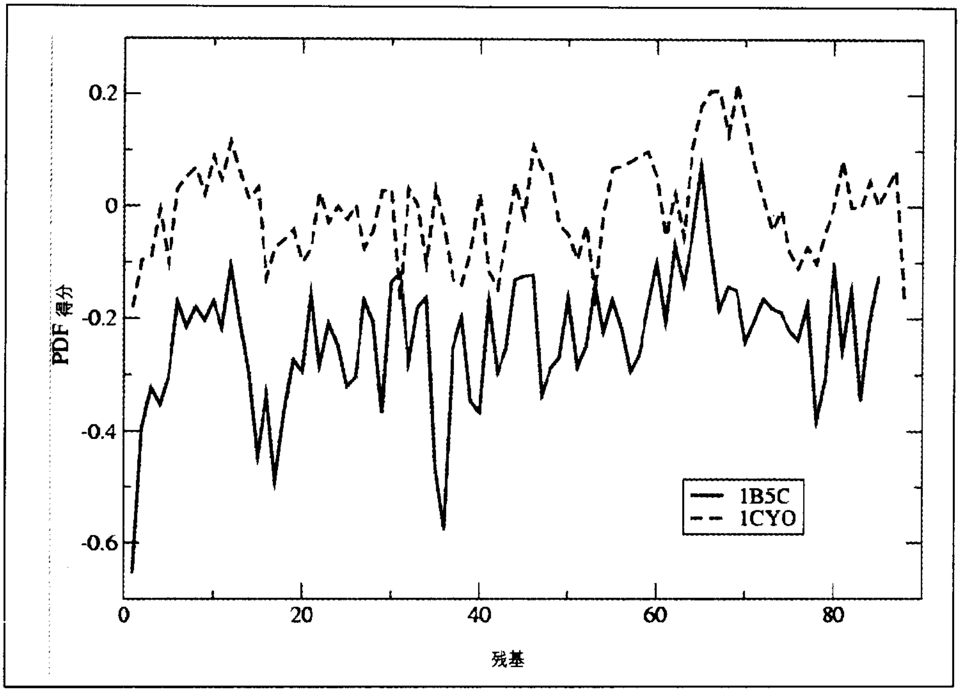


图9-16　过时的 PDB 结构（1B5C）和代替它的（1CYO）的 PDF 得分比较

蛋白质资源数据库

最近 Web上出现了几个包括蛋白质结构和功能的新数据库，它们是为基因组水平信息的用户设计的。最著名的一些是 GeneCensus、PRESAGE 和 BIND。这些数据库仍然处于相对次要的地位，还不能代替 PDB 和 NCBI在生物学研究中的中心地位。然而，就是这些和其他的类似资源不久将会变得对分子生物学研究至关重要。

GeneCensus

GeneCensus计划是一个几个基因组的蛋白质内容的广泛的基于序列的比较。在写这部分时，GeneCensus 包括 20个基因组的信息。目前， GeneCensus 可以使用 PDB ID 或者 ORF ID 搜索来定位 GeneCensus 基因组中专一蛋白质折叠的存在。

PRESAGE

PRESAGE 是有关各种蛋白质结构的实验进展信息的数据库。可以使用 TIGR ORF 编码，NCBI 或者 SWISS-PROT 登录号和许多其他编码号搜索 PRESAGE，找出是否有人正在试图分离、结晶和解析某个蛋白质的结构，如果有，他们还有多少事情要做。在本书出版时 PRESAGE 还相对较新；目前它只包含 6000 条记录，还不能保证很全面。它还不能够直接使用 BLAST 或者 FASTA 对一个序列进行搜索，因此在对一个未知序列进行 PRESAGE 检查之前，不得不先搜索匹配的登录号。然而，原则上，PRESAGE 数据库有望对晶体学家和他们的合作者，以及对于想知道他们的蛋白质结构是否已被解析的好奇的分子生物学社团的成员有帮助。

BIND

BIND（Biomolecular Interaction Network Database，分子生物学相互作用网络数据库）是另一个相对较新的数据仓库，它是由 Samuel Lunenfeld 研究所提供的。BIND 的设计目的是作为有关已知大分子相互作用信息的中心放置点。BIND 是一个复杂的数据库，它包括数据库中物质间的相互作用信息、分子化合物信息和代谢途径信息。BIND 格式设计为包括存储在数据库中的用于观测相互作用的实验条件的信息，以及结合点位置、生化活性、动力学和热动力学的信息。 BIND 仍处于 β 测试阶段，只包括几百个相互作用。然而， BIND 已经投资雇佣编制索引的人，预计在不久的将来会快速增长，BIND 数据输入过程的一个有趣的方面是，已实现自动阅读存在的期刊文献的方法，以从存放在很不方便的图书堆中将已知的相互作用提取出来，并把它们放在更有效的公共域。

把一切结合在一起

我们不能给你一个使用本章描述的技术来研究蛋白质性质的简单秘诀。从系统到系统有太多的差异，作为一个生物学家，你想知道的东西有太多的变化。然而，结构建模方法也有一些共同的特征，包括：

> 通过学习收集关于系统的有用的结构和生物物理信息。从定点突变到传统生化和生物物理研究的一切都可能是有用的。

- 使用多序列分析来分析作为相关家族一部分的蛋白质。这可能会使你深入理解重要功能残基和活性位点的位置。
- 分析晶体结构或者理论模型来确定包埋的极性和带电荷的残基、特殊的氢键、结构化的溶剂分子网和其他可能涉及结构稳定性或者功能的化学特征。
- 通过叠加或者比较一个相关蛋白质家族来分析它们以确定共同的特征。
- 将确定的性质（如果突变就会影响功能的已知位点、多个序列的保守位点，等等）绘制到蛋白质序列和结构中。
- 将结构可视化，解释重要的氨基酸和位点的潜在位置。
- 计算分子表面和通过它们的形状研究可能的底物结合位点或分子相互作用区域的性质。
- 计算静电势能并建立诸如单个氨基酸pKa 值或者分子相互作用能等静电性质的模型。

特别强的静电势能或者特殊的 pKa 值可能指示具有催化重要性的区域。

很明显，这种类型的分析需要真正理解蛋白质化学。我们为你确定了结构生物学的工具，但是你要决定如何把它们结合在一起使用。为了帮助你达到目的，表 9-3 包括分子结构工具的快速参考和通常如何使用它们的方法。

表 9-3　结构工具和技术

目的	原因	可用工具
查看分子结构	计算机图形是详细查看蛋白质结构的惟一方法	浏览器插件：RasMol、Cn3D、SWISS-PDBViewer，独立的：MolMol，MidasPlus、VMD
建立蛋白质的高质量 PostScript 示意图和彩色图谱	用于出版	MolScript
建立活性位点示意图	帮助确定功能位点的结构组分；用于出版	LIGPLOT

表9-3　结构工具和技术（续）

目的	原因	可用工具
结构分类	确定蛋白质间的关系	CATH，SCOP
二级结构分析	在二级结构水平上提取可识别的特征，可以帮助分类	DSSP，STRIDE
拓扑学分析	提取可识别超二级基序，可以帮助分类	TOPS
结构域确定	提取可识别结构域，可以帮助分类	3Dee
惟一结构数据库子集	为基于知识的建模消除源数据集合的偏差	PDBSelect，精选的PDB数据库
结构比对	确定远相关蛋白质间的关系，因为进化，可能不再有可识别的序列相似性，但是有保守的结构相似性	CE，DALI，VAST
分子几何分析	确定结构模型中受限制的构象或者错误的表示区域	PROCHECK，WHAT IF
分子内接触分析	确定残基－残基间的相互作用，可以帮助确定活性位点、结构稳定性特征，等等。	CSU，HBPLUS
溶剂可接近性计算	确定和溶剂相互作用的氨基酸	naccess，Alpha Shapes
溶剂建模	在分子周围放置一个实际的化学溶剂壳，为一些模拟做准备；帮助理解功能机制	HBUILD
分子表面可视化	可以得到对分子形状和化学表面性质的可视化理解	GRASP，GRASS服务器，SWISS-PDBViewer
静电势能计算	可视化蛋白质重要的化学表面特征，作为pKa计算、结合能计算和Brownian动力学模拟的第一步	UHBD，DelPhi
蛋白质pKa计算	建立蛋白质pH依赖性行为的模型，确定处于可能的活性位点和确定特殊化学环境中的残基	UHBD，DelPhi

第十章

根据序列预测蛋白质的结构和功能

在蛋白质的氨基酸序列中包含着它本身和内在的令人感兴趣的信息。如果可能的话，蛋白质序列可以通过和其他蛋白质序列的对比来建立它同已知蛋白质家族的关系，并且提供关于生物化学功能进化的信息。然而，对于理解蛋白质功能这个目的，蛋白质的三维结构比它的序列更有用。

允许蛋白质行使各种生物化学功能的关键性质是蛋白质链中的氨基酸组成，它以某种方式惟一地决定了蛋白质的三维结构（注 1)。根据 20种氨基酸的概率，即使是很短的蛋白质序列，它们也会有巨大数量的组合方式，这意味着给定时间，生物实际上可以使蛋白质向着任何可以想像的方向进化。

每当一个特别的蛋白质链在细胞中合成的时候，它通过折叠（fold）使行使特殊蛋白质功能的每一个重要化学集团形成精确的几何排列。由蛋白质序列所决定的折叠是不变的。某个特定的蛋白质每次总是精确地折叠成同样的结构。

尽管在蛋白质中有这种一致性，但是还没有人提出如何仅仅根据蛋白质序列准确预测蛋白质将要折叠成的三维结构。蛋白质中氨基酸序列的模式是明显存在的，但是这些模式是简并性的（degenerate)，也就是说，不止一个序列能够特异性地形成某类特殊的折叠。虽然氨基酸组合形成一种特定长度序列的方式有成千上万种，但是

注 1: 顺便提及，“以某种方式”代表几百个研究者几十年来对一个基本问题的研究工作，直到今天这个工作还没有完成。

蛋白质结构组织自身的独特方式看起来非常少。在PDB 中只可以看到几百种折叠方式。几乎完全没有同源性序列的蛋白质可以折叠成相似的结构。因此，从序列预测蛋白质的结构是件困难的事情。

蛋白质结构的确定

如果能够用实验方法确定蛋白质的结构，结构预测又是如此艰苦，为什么还非要预测蛋白质结构不可呢？答案是解析蛋白质结构很困难，在实验过程中存在着许多瓶颈。虽然第一个蛋白质结构比第一个 DNA 序列早 20 年确定。但是在这期间，蛋白质结构数据库的增长速度远低于序列数据库。现在在 PDB 中的蛋白质结构有 10000 种，而 GenBank 中的基因序列有 10000000 条。只有大约 3000 种独特的蛋白质结构已经被解析（序列同源性超过 95% 的蛋白质除外）。其中大约有 1000条是来源于彼此序列非常不同的蛋白质（序列同源性不超过 25% 的蛋白质）。

用 X 射线晶体学解析蛋白质结构

在 19 世纪 30 年代末，已经知道蛋白质由氨基酸组成，虽然还没有证明这些组分是结合在一起形成特殊的序列。Linus Pauling 和 Robert Corey 开始使用 X射线晶体学来研究氨基酸和多肽的原子结构。

当 Pauling 和 Corey开始他们的实验时，纯的蛋白质已经可以结晶。然而， X射线晶体学需要大的、纯净的蛋白质晶体。蛋白质的纯化和结晶技术还没有发展到产生有用的晶体的时候。Pauling和 Corney在氨基酸和多肽的研究中发现的是肽键是平的、刚性的，羧基氧和氨基氢几乎总是处于肽键的两侧（参见图 9-1 的肽键示意图)。使用这些信息来限制模型，结合在氨基酸中观察到的原子键长和键角， Pauling 和 Corney建立了肽链的结构模型。结果他们提出了在蛋白质中的两类重复结构：α螺旋和 β片层，如图 9-5 所示。

在开始于 20世纪 50年代早期的试验中，John Kendrew确定了一个叫作血球素的蛋白结构，Max Perutz确定了一个叫血红素的类似的蛋白质结构。两个蛋白都是氧的运输体，很容易从血液中大量提取并结晶。如果没有现代计算机的帮助，要得到合适的高质量的 X 射线数据，并分析数据，需要几年的时间。血球素和血红素的结构被发现是由高密度的棒组成的，这正是由 Pauling 提出的 α螺旋所预计的尺度。两

年以后，一组更高质量的晶体衍射数据精确确定了血红素 1260 个原子中的 1200 个允许的位置。Kendrew和 Perutz的实验开启了其他蛋白质 X射线晶体衍射分析之门。

在 X射线晶体学中，物质的晶体被置于 X射线的光束中，X射线被晶体中原子周围的电子云反射。在一个蛋白质晶体中，单个的蛋白质分子被安置在有规律的晶格中，因此 X射线被晶体以一种有规律的模式反射。被蛋白质晶体散射回来的 X射线反射光（reflection）可以被分析产生蛋白质的电子密度图（见图 10-1，该图由威斯康星大学 Madison 学院的 Holden小组和Bruker Analytical X Ray Systems 提供）。蛋白质原子的坐标通过为组成已知蛋白质序列的原子建立最适合的电子密度的模型来产生。匹配过程不是明确的；有许多不同的方式可以使原子结构和电子的密度图相匹配，但是并不是所有都是在化学上正确的。蛋白质结构不是晶体中原子位置的精确表示；它仅仅是最符合蛋白质电子云密度和控制蛋白质结构的立体化学限制的模型。

为了用 X射线晶体学确定蛋白质的结构，需要极纯的蛋白质样品。蛋白质样品必须形成相对较大（大约 0.5毫米）、单个的、无污染的晶体。利用基因重组技术生产大量的纯的蛋白质样品变得更容易。然而，结晶蛋白质在一定程度上仍然是一门艺术。对使蛋白质快速结晶的条件还没有统一的秘诀（如，蛋白质溶液的盐组分和 pH值），甚至是在发现了正确的溶液条件后，形成适合大小的晶体也可能要花费几个月的时间。

许多蛋白质结构是根本不能形成晶体的。例如，那些在细胞膜中行使功能的蛋白质在水中通常是不溶的，并且在溶液中倾向于聚合，因此，通过新 X射线晶体学解析膜蛋白质的结构十分困难。膜整合蛋白质大约占生物总蛋白质（蛋白质组，*proteome*）的 30%，但是只有少于一打的这类蛋白质可结晶到足够纯以形成结构（可以原子分辨率解析）。

通过 NMR 光谱解析结构

通过 NMR （核磁共振）光谱解析的蛋白质结构数目正在增加。图 10-2 显示了来自NMR光谱的原始数据。NMR 使用非零自旋探测原子核；这些原子核产生的信号依靠于它们的电子环境在磁场中漂移（shift）。通过解释分子的 NMR 光谱中观察到的化学漂移，可以推测分子中特定原子间的距离（图 10-2 的图像是由 Bruker Analytical X Ray Systems的 Clemens Anklin 提供）。

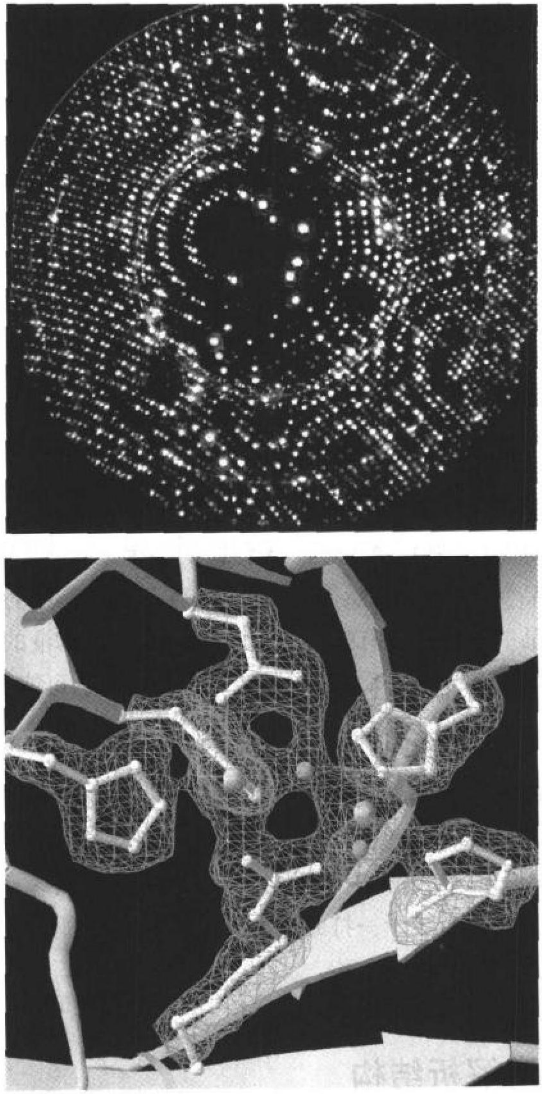


图 10-1　含锌磷酸二酯酶三维结构的 X 射线衍射模式和相应的电子密度图

使用NMR研究蛋白质，蛋白质必须足够小，足以在溶液中快速转动（分子量在30KD以内），蛋白质必须是高度可溶的，并且在室温下可以长达几天保持稳定性。

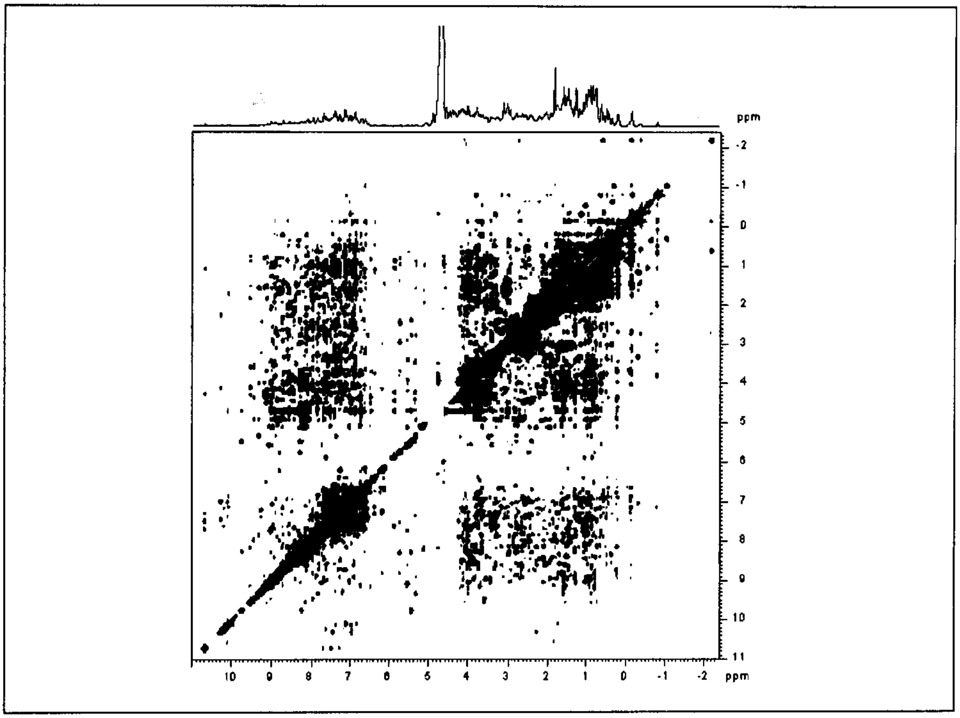


图 10-2 溶菌酶的 NOESY（二维 NMR）图谱

对NMR实验得到的化学漂移数据进行分析可产生一组蛋白质标记原子的距离限制。解析一个 NMR结构意味着产生一个或一组模型，它们满足 NMR 实验决定的所有已知距离限制和掌握蛋白质结构的一般立体化学限制。NMR模型通常是以20~40个模型为一组释放的，因为NMR所确定的结构比晶体学所确定的结构还要模糊。NMR的平均结构是通过平均一组模型得到的（见图 10-3)。依赖于这种平均是如何进行的，立体化学错误可能会被引进最终的结构中，因此在使用它们建模之前，检查平均结构的质量通常是明智的。

预测蛋白质的结构

理论上，在蛋白质分析中要做的是提取蛋白质的序列，这在实验中很容易得到，然而预测蛋白质的结构是昂贵的，甚至有时在实验中无法确定。下面这些方面也是令人感兴趣的：从序列准确预测功能，确定未定性的三维结构的功能位点，最终合成



图 10-3　一组 NMR 模型的结构差异

设计的蛋白质 —— 可以做我们需要它们做的任何事情的分子机器。但是不了解序列是如何决定结构的，这些其他目标就不能真正实现。

在蛋白质的计算建模中有两种方法。第一种是基于知识的建模。基于知识的方法使用从已存在的结构数据库中得到的参数来评价和优化结构或从序列预测结构（蛋白质结构预测问题）。第二种方法是基于物理力和分子动力学的模拟。物理化学模拟常常用于试图建立模型解决蛋白质是如何从无功能的、非紧密的变性（denatured）状态折叠成紧密的、有功能的天然（nature）状态（蛋白质折叠问题）。在本章中，我们将着眼于基于知识的蛋白质结构预测和分析方法，生物信息学在其中起到了重要作用。

从蛋白质序列从头开始预测蛋白质的结构在计算生物学中还有一些未解决的问题。虽然许多学者已经开始开发结构预测的方法，但是产生大量成功三维结构预测的惟一方法是基于结构同源性的方法。如果有一个蛋白质序列并想知道其结构，你或者需要一个同源性结构作为建模的模板，或者需要找到一个晶体学家和你合作用实验方法解析结构。

蛋白质结构社区以两种方式表示序列和结构间数据库的缝隙。结构基因组学的先驱计划（努力用实验方法用整个基因组解决所有或大部分蛋白质的结构的编码）正在

几个研究所进行。然而，这些计划很少有机会跟上蛋白质数据库日益增长的容量，至少在近期是如此。

CASP: 寻找圣杯

考虑到数据库缝隙，计算结构预测是个热点。它常被称为计算生物学的“圣杯（holy grail)”；它是一个重要的目标，也是一个难以捉摸的，或许是不可能的目标。然而，从文献中跟踪蛋白质结构预测领域里的进展和找出已经部分成功的方法是可能的。

每两年时间，结构预测研究组织就会举行CASP（Community Wide Experiment on the Critical Assessment of Techniques for Protein Structure Prediction，关于蛋白质结构预测的重要预测技术的广泛试验社区比赛， *http://predictioncenter.llnl.gov/*）。CASP的比赛结果显示了蛋白质结构预测的最新方法。CASP有三个竞争领域：同源性建模、穿线法和从头预测。而且， CASP 是评估结构预测正确性的新方法的试验地。

同源性建模（homology modeling）着眼于使用从已知序列得到的结构模板来构建蛋白质的全原子模型。1998年CASP的预测质量显示，基于同源性的结构预测对于任何可以找到具有显著性同源性结构的实例在产生合理的模型方面是成功的。对同源性预测的挑战和序列比对一样是在 Twilight Zone区 ——低于 25%序列同源性的区域，发现有意义的序列同源性。

穿线法（threading method）是取未定性蛋白质结构的氨基酸序列，快速计算出基于已存在的三维结构的模型，然后评估这些模型来决定未知氨基酸序列对每一个模板结构的适合程度。在最近CASP比赛中报道的所有穿线法中，有少于一半的情况可以产生准确的结构模型。用于发现不能用标准的序列比对发现的远同源性，该方法更成功。如果序列的部分和一个折叠很吻合，通常会产生一个比对，虽然可能没有足够的信息建立完整的模型。

从头开始预测（ab-initio prediction）的方法着眼于在没有预先的信息的情况下创建结构。然而，在CASP评估的从头开始方法中，没有一个可以产生可靠的准确的模型，各种方法在这个领域中显示了一些希望。一个基于信息学的结构预测战略已开发了短结构片段的表示数据库，从这些结构片段中可以建立结构。因为不在片段库中的结构“单词”是超过限制的，因此要在建模中搜索的结构空间也会受到某些限

制。另一个常见的从头开始的方法是使用简化的蛋白质结构代表来模拟折叠。在这些方法中，蛋白质可以表示为一条线上的珠子。在一些方法中，每一个氨基酸或者固定的二级结构变成具有某种性质的珠子，它们可以吸引或者排斥其他类型的珠子，统计学机制模拟方法用于寻找构象空间可能的简化模型。这些方法在确定蛋白质折叠中可能成功，甚至是在结构的细节不能被预测的时候。

从三维到一维

蛋白质和DNA实际上是由成千上百万个原子组成的复杂的化学结构。即使是使用现代计算机，精确地模拟这样复杂的结构也是不可能的，因此计算生物学家在开发分析方法时使用了蛋白质和DNA的抽象(adstraction)表示法。最常用的生物大分子抽象表示法是单字母序列。然而，把复杂结构的信息内容减少到简单的序列编码会丢失一些有用的信息。

第七章和第八章讨论的序列分析方法把蛋白质作为字符串。为了进行序列比较，蛋白质的字符序列差不多可以完全代表蛋白质结构。然而，甚至是记分蛋白质序列比对中的替换矩阵也需要指向蛋白质更复杂的化学性质。一些氨基酸彼此在化学上是相似的。可以相互代替。一些是不同的，有些大，有些小，有些是极性的，有些是非极性的。替换矩阵是简单的定量方式，它把氨基酸的性质信息转化成线性序列。不管特性有多相似，替换矩阵反映了将一个氨基酸替换成为另一个氨基酸的倾向性，但是仅此而已。

然而，蛋白质结构中的每一个氨基酸（或者DNA或RNA中的每一个碱基，如我们开始学到的那样）不仅仅存在于它的序列上下文关系中。一维信息还不足以清晰地显示蛋白质结构和功能是如何决定于序列的。残基的三维结构和化学上下文关系包括许多种类型的信息。

直到最近，蛋白质结构的三维信息通常通过从三维到一维的映射，浓缩成更容易分析的形式。从数据库中得到的性质通常以对序列每一个残基的数字或字母字符得分的形式来映射。基于知识的二级结构预测方法是从三维到一维映射的最初应用之一。通过映射二级结构上下文关系到序列信息中作为性质（把代表“螺旋”、“片层”,或者“卷曲”的编码附加给每一个氨基酸）可以从结构数据库中得到一套二级结构的

倾向（secondary structure propensity），然后用它们预测新的序列的二级结构的内容。

除了二级结构外，蛋白质序列中的每一个氨基酸还有什么重要的信息呢？一些常用的性质是溶剂可接近性、酸碱性、极性、最近序列邻居和最近空间邻居。与第九章中讨论的一样，所有这些性质都与分子间的相互作用有关。

蛋白质序列中的特征检测

蛋白质序列分析是以对蛋白质链化学组分的物理化学性质的了解程度和对蛋白质结构及亚结构中特殊位置的特定氨基酸的出现频率的了解程度为基础的。虽然蛋白质序列分析工具是对一维序列数据进行操作，但是它们包含关于结构特征如何映射为序列数据的隐式假设。在使用这些工具之前，最好了解一些蛋白质生物化学的知识。

蛋白质序列的特征代表蛋白质功能的细节。这些通常包括翻译后修饰位点和定位信号。翻译后修饰（post-translational modification）是蛋白质从信使 RNA 翻译出来后的化学改变。包括蛋白质的截短（切割）和调节蛋白质行为的化学集团的添加（磷酸化、糖基化、乙酰化是常见的例子）。定位(localization）或者靶向信号(targeting signal）由细胞用来保证蛋白质在正确的时间出现在正确的地方。包括核定位信号、ER 靶肽和跨模螺旋（我们在第九章已经看到过）。

蛋白质序列特征检测往往是分子生物学家处理计算化学时遇到的第一个问题。不幸的是，用于特征检测的软件工具没有被定位到中心位置。它们往往分散在个人和实验室的网页上，为了找到它们，需要用你喜欢的搜索引擎做些挖掘工作。

一个基于 Web的蛋白质序列特征检测资源集合是 Technical University of Demark's Center(丹麦中心技术大学)维护的 Bio-logical Sequence Analysis Prediction Server (生物序列分析预测服务器，*http://www.cbs.dtu.dk/services/*)。这个站点提供了对（在其他事情中）切割位点、糖基化位点和亚细胞定位信号的基于服务器的程序的访问。这些工具的工作原理十分相似；它们搜索已知与各种翻译后修饰有关的简单的序列模式或简图。这些程序有标准的界面，可以直接使用：每一个都有提交表单，你可以把感兴趣的序列粘贴进去，每一个程序都会返回一个包含结果的 HTML页。

二级结构预测

二级结构预测常常被认为是预测蛋白质结构的第一步。在使用任何预测方法时，在将这些程序的输出解释为二级结构的实际预测时，可在一定程度上使用怀疑论。二级结构预测可以帮助我们分析和比较蛋白质结构。在本节中，我们将简单探索二级结构预测的方法、它们的测量方法及一些应用。

蛋白质二级结构预测是根据它们被预测的局部结构，对蛋白质序列中的氨基酸进行分类。二级结构通常分为三类之一（α螺旋、β片层或者卷曲），虽然状态的数目确定于使用的模型。

二级结构预测方法可以分为基于比对和基于单个序列的方法。基于比对的（alignment-based）二级结构预测是非常直观的，与序列的简图概念有关。在基于比对的二级结构预测中，调查者可找到和未知序列相似的序列家族。然后假设序列家族中的同源区有同样的二级结构，预测不是基于一个序列而是一组序列中的所有序列的一致序列。另一方面，基于单序列的（single sequence-based）方法只为一个未知序列预测局部结构。

基于比对的方法和杂合方法

二级结构预测的现代方法使用多序列比对的信息，或者多个方法的结合，或者同时使用上述两者。这些方法声称准确度为 70%~77%。这些程序中的许多可以在 Web 上使用。它们用一个序列（通常是 FASTA 格式）作为输入，对它们执行一些过程，然后返回一个预测，通常是通过邮件，因为预测算法是需要大量计算的，而且倾向于在队列中计算。下面是最常用的六个方法的列表：

PHD

PHD 结合了许多神经网络的结果，每一个结果都是根据局部序列上下文关系和整体蛋白质性质（蛋白质长度、氨基酸频率等）来预测残基的二级结构。那么，最终的预测是这些神经网络每一个输出的算术平均值。这种结合方案被称为陪审团决定法（jury decision）或者（更通俗的）称为所有胜利者（winner-take-all）法。PHD 被认为是二级结构预测的标准。

PSIPRED

PSIPRED结合了神经网络预测和来源于 PSI-BLAST 数据库搜索的多序列比对。PSIPRED是 CASP 3 上最好的二级结构预测方法之一。

JPred

JPred二级结构预测采纳其他几个互补的预测方法的一致性结果，并补充以多序列比对信息。JPred是另一个最好的二级结构预测方法 。JPred 服务器返回的结果可以使用 Jalview 比对编辑器依次显示 、编辑和保存用于其他程序。

PREDATOR

PREDATOR 结合多序列比对信息和氨基酸的氢键性质来预测二级结构。

PSA

PSA 是基于 Markov 模型的另一个二级结构预测方法。它因为详细的图形输出而受人关注，其中图形代表蛋白质序列中每一个位置的螺旋、片层和卷曲的预测概率。

单序列预测方法

在 20 世纪 70 年代第一个结构预测的方法是单序列方法。 Chou-Fasman方法使用氨基酸的物理化学数据中派生出的规律来预测二级结构。 GOR 算法（因其作者 Garnier、Osguthorpe 和 Robson 而得名）及其后继者使用已知结构的蛋白质中氨基酸在螺旋、片层和卷曲中出现的频率信息来预测结构。两种方法都还在使用，通常是在诸如 SDSC生物学工作台等多工具服务器上使用。现在的基于结构规律和频率的方法可以达到 70%~75% 的准确性，尤其是当相关序列家族而不是单序列已经被分析时。

20世纪80年代人工智能方法的流行浪潮促进了基于人工智能方法的二级结构预测，最著名的模式识别方法是（旧金山，加利福尼亚大学） Fred Cohen 实验室开发的方法，（Imperial Cancer Research Fund ，皇家癌症研究基金会的） Michael Sternberg 方法和（ Johns Hopkins）的 Terrence Sejnowski及 N. Qian 的神经网络应用法。这些方法使用人工智能技术自动进行知识的获取和应用，开发与早期的单序列方法相似的信息。

测量预测的准确性

发表二级结构预测方法的作者常常使用称作 Q_3 的预测准确性的测量方法。Q_3的得分定义为：

Q = 真阳性 + 真阴性／总氨基酸数

第二个预测准确性的测量方法是片段重叠得分（Sov，segment overlap score)，是由 Burkhard Rost 及其同事提出的。Sov 度量倾向于比 Q_3 更严谨，因为它给单一类型二级结构的非重叠区的分值很高，而对稀有预测进行罚分（见图 10-4)。

a) 好的预测

```
HHHHHHHHHHHEEE         EEEHHHHHHHHHHHHHHHHHHHHH    EEEE
TEAMRNDIIARICGEDGQVDPNCFVLAQSIVFSAMEQEHFSEFLRSHHFCKYQIE
```

a) 稀有的预测

```
H H  H H HHE E  E      E E H H HH H   H H  H HH    E  E
TEAMRNDIIARICGEDGQVDPNCFVLAQSIVFSAMEQEHFSEFLRSHHFCKYQIE
```

图 10-4　好的和坏的（稀有的）二级结构预测

在比较方法时是值得保守的；看它们的平均得分和标准差，而不是最好报告的得分。正如你看到的，Q_3和 Sov是相当简单的统计学。与序列比较中的 E值不同，也不是一种测试统计；两种方法均测量正确预测残基的百分数，而不是测量预测准确性的显著性。结合基因寻找器，确信你知道何种数据用于考察预测方法。

预测的应用

起初，预测二级结构的目的是为了使预测尽可能精确。许多作者将此作为他们的目标，导致结果像一团乱麻。与前面提到的一样，二级结构预测最难攻克的地方是它并不十分准确。然而，二级结构预测方法在生物信息学上已被实际应用，尤其在检测远的同源性时。药物公司在药物靶点研究中通过比较二级结构的预测定位潜在的远同源性。被预测的二级结构模式能够预测蛋白质折叠的分类并在结构基因组学中选择靶点。

PredictProtein和 JPred等二级结构预测工具结合了几种方法的预测结构，包括二级结构预测和穿线法。使用几种互补方法（单序列和基于同源性的方法）预测二级结构，比使用单一方法可得到更好的结果。如果所有的方法对一个区域的预测结构一致，你可以比用一两个程序得到一致的结果更加相信预测的结果，这称为机器学习中的投票（voting）程序。

与其他预测一样，如果二级结构的一些信息是已知的，二级结构预测会更有用。例如，即使蛋白质中的一段短结构已被确定，这些数据也可以用作预测的公正检查。

预测跨膜螺旋

跨膜螺旋预测（transmembrane helix prediction）与二级结构预测相关，涉及蛋白质序列中可插入细胞膜的区域的识别。预测蛋白质序列中跨膜螺旋的方法需要鉴定序列中可以折叠成螺旋并存在于膜疏水环境中的区域。跨膜螺旋预测在 20世纪 80年代初超出了对疏水性的研究，由（旧金山，加利福尼亚大学的） Russell Doolittle 领导。而且在网上可以找到许多跨膜螺旋预测的服务器。用作标准的跨膜螺旋预测程序包括 TMHMM（*http://www/cbs.dtu.dk/services/TMHMM-1.0*），MEMSAT（*http://insulin.brunel.ac.uk/~jones/memsat.html*）和 TopPred（*http://www.sbc.su.se/~erikw/toppred2/*）。

虽然可溶性蛋白质的结构确定可能是困难的，但是更大的挑战来自于膜结合蛋白质结构的确定。一些最有意思的生物学过程都与膜结合蛋白质相关。例如，光合作用、视觉、神经刺激、呼吸作用、免疫反应和信号从一个细胞传递给另一个细胞。可是只有少数膜蛋白质已结晶化，因为这些蛋白质不是完全存在于水溶液中，它们的物理化学性质和可溶性蛋白质非常不同，它们要在非常规的条件下结晶——如果它们确实可以结晶的话。

结果是许多检测蛋白质序列中的跨膜片段的计算机程序的出现。这些序列具有明显的特征，因而可合理地检测到。为了跨过细胞膜，α螺旋必须有大约 17~25个氨基酸的长度。因为细胞膜的内部是由脂肪酸的长碳氢链组成，所以膜中的α螺旋必须存在相对的面向膜的非极性面才能在能量上是有利的。

早期的跨膜片段鉴定程序会直接探讨这些问题。通过分析氨基酸序列的每 17~25个残基窗口，对每一个窗口产生一个疏水性（hydrophobicity）得分，得分高的片段就

被预测为跨膜螺旋。对这些方法的最近改进将单个跨膜片段的预测准确性提高到了90%~95%。

膜中蛋白质的拓扑结构不易预测。膜中第一个螺旋的方向决定了所有剩余螺旋的方向。螺旋的连接可按照在膜的一侧或者另一侧来分类，但是在生理学上确定哪一侧是哪个更加复杂。

穿线法

穿线法结构分析的基本原理是试着把未知的氨基酸序列和各种已存的三维结构相匹配，并评估序列折叠成那种结构的合适度。所有的穿线法操作都是基于这种假设，但是它们具体的细节有所不同。

穿线法不是建立蛋白质所有原子的精细模型；而是快速用未知序列的氨基酸侧链替换已知序列中的氨基酸位置。折叠蛋白质中的每一个氨基酸侧链位置都可以根据环境来描述：侧链暴露在溶剂中的程度，如果它不是暴露在溶剂中，它和其他哪些氨基酸有联系。如果疏水氨基酸残基处于溶剂不可接近的环境中或者亲水残基处于蛋白质的表面，穿线模型的得分会很高。但是只有包埋的电荷和极性集团处于合适的电荷作用或者氢键模式时，这些高得分才是可能的。

穿线法最适合用于折叠的识别，而不是模型的建立。为了这个目的， UCLA-DOE 结构预测服务器（*http://www.doembi.ucla.edu/people/frsur.html*） 是最容易使用的工具。它允许你提交单一的序列，尝试多个折叠识别和评估方法，包括 DOE实验室自己的 Gon 预测方法，EMBL 的 TOPITS 方法和 NIH的 123D+ 方法。其他特征，包括 BLAST 和 FASTA 搜索、PROSITE搜索，以及评价序列和它的明显结构的适合度的 Bowie 简图都是可以得到的。

另一个穿线法服务器是由 Biomolecular Modelling Laboratory of the Imeperial Cancer Research Fund （皇家癌症研究基金的分子建模实验室）提供的 3D-PSSM 服务器，它提供了基于未知序列家族的简图的折叠预测方法。3D-PSSM将多个分析步骤组合到一个简单的界面中。首先，使用 PSI-BLAST 在一个非冗余蛋白质序列数据库中比较未知蛋白质序列，产生该蛋白质的位置特异性得分矩阵（PSSM）。用 PSSM 查询比较数据库中的所有序列。查询序列也比较数据库中所有蛋白质家族的 1D-PSSM （基于标准的多序列比对）和 3D-PSSM （基于结构比对）。基于这些比较的二级结构

预测被显示和排列成与查询蛋白质可能结构匹配的已知二级结构。从 3D-PSSM 搜索得到的结果以一种易于理解的图形格式表示，但是它们也可以和未知序列的 α碳模型一起下载，后者是使用每一个可能的结构匹配作为模板建立的。

大多数穿线法被认为是实验性的，新方法总是处于发展之中。可用不止一种方法来帮助鉴定任何一个未知序列，结果可以用多个方法的一致性解释。对于使用穿线服务器建立的任何结构模型，要记住的最主要的事情是：它可能缺少原子细节，也可能是建立在轻微或者是严重的错误排列上的。穿线法的设计目的是假设序列作为可能的候选者去匹配特殊的折叠，而不是建立有用的模型。利用穿线服务器产生的假设的结构比对可以作为同源性建模的基础，但是在建立全原子模型之前，它们应该经过仔细的检查和编辑。

三维结构预测

像本章开始提到的那样，从序列预测蛋白质的结构是件十分困难的任务，还没有一个可以满足各种参数的方法，可是有许多工具可用来预测三维结构。它们可以分成两类：同源性建模和从头开始的预测。

同源性建模

假设对着已知结构的另一个蛋白质序列来排列一个蛋白质的序列（“靶”序列）。如果靶序列和已知结构序列有很高的相似性（在整个序列的全长），在合理的信任度上，我们可以使用已知结构作为靶蛋白质的模板。

这里有一个用于同源性建模的标准过程。虽然执行每一步骤的程序可能不同，但是过程是不变的：

1. 使用未知序列作为查询来搜索已知蛋白质结构。
2. 产生未知序列和模板序列最可能的完整比对。
3. 以模板结构骨架作为模型，建立蛋白质骨架模型。
4. 在靶序列或者模板序列的有空位区域，使用环建模过程代替合适长度的片段。
5. 给骨架模型加上侧链。

6. 优化侧链的位置。

7. 使用能量最小和已知的优化知识来优化结构。

同源性建模方案成功的关键通常不是建模使用的软件或者服务器。在设计与模板结构好的比对时的技巧更重要。像我们在本章结尾讨论的例子一样，结合标准的序列比对方法、简图方法和结构比对技术，可以产生这种比对。一旦有一个好的比对存在，就有几个程序可以用这个比对的信息产生结构模型。

Modeller

Modeller（*http://guitar.rockefeller.edu/modeller/modeller.html*）是一个同源性建模的程序，它可以作为一个单独的程序或者作为 MSI 的 Quanta 软件包（*http://www.msi.com*）的一部分在学术上自由使用。

Modeller没有自己的图形界面，但是一旦你适应了命令行环境，它在使用上根本不存在困难。Modeller的可执行程序可从 Rockefeller大学的网站下载，按照发布包中的 *README*文件的提示，安装很简单。每一种操作系统都有几种不同的可执行程序，应该根据使用它们建模问题的大小来选择。*README*文件包括各种可执行程序的阵列大小限制信息。包括对总原子数、总残基数和输入比对的总序列数的限制。

当对 Modeller进行输入的时候，需要两个输入文件：一个比对文件和一个 Modeller 脚本文件。比对文件格式在 Modeller 的主页上有详细描述；一个简单比对的 Modeller 脚本文件仅仅包括由 TOP语言（Modeller 的内部语言）写成的几行。Modeller 可以为一个输入计算出多个模型。如果脚本文件例子显示的 *ENDING_MODEL*值被设为 1以外的其他值，则会得到多个模型。通常，我们愿意得到多个模型，独立地评价每一个模型，选择最好的结果作为最终模型。

Modeller文档提供的例子显示了使用一个已知蛋白质和未知序列间预产生的比对来建立比对：

```
INCLUDE                                      # 包含预定义的 TOP 例程
SET ALNFILE = 'alignment.ali'                # 比对文件名
SET KNOWNS  = '5fd1'                         # 模板代码
SET SEQUENCE = '1fdx'                        # 目标代码
SET ATOMS_FILES_DIRECTORY = './:../atom_files'# 输入原子文件的目录
SET STARTING_MODEL= 1                        # 第一个模型的索引
SET ENDING_MODEL = 1                         # 下一个模型的索引
```

```
                                         # （决定要计算多少个模型）
    CALL ROUTINE = 'model'               # 进行同源性建模
```

Modeller 使用 *mod scriptname* 命令运行。如果命名脚本文件为 *fdx.top*， 则命令为 *mod fdx*。

Modeller 具有多个功能，并具有帮助你准备输入文件的内置命令：

SEQUENCE_SEARCH
　　在折叠分类代表结构中搜索相似序列。

MALIGN3D
　　比对两个或更多结构。

ALIGN
　　比对两个序列块。

CHECK ALIGNMENT
　　评估建模用比对。

COMPARE_SEQUENCES
　　给基于成对相似性的比对中的序列打分。

SUPERPOSE
　　将一个模型和另一个模型或模板结构叠加。

ENERGY
　　在模型结构中产生限制性偏差的报告。

像前面的脚本样例显示的那样，每一条命令需要通过调用该命令的脚本提交给Modeller程序。几十个其他 Modeller命令和书写脚本的细节在Modeller手册中都有描述。

自动同源性建模的一个中止警告，可能是侧链原子不能正确定位到产生的模型中；模型的自动建立方法着眼于建立合理的蛋白质结构骨架模型，因为同源性提供了可以相信的这方面的信息。同源性不会提供侧链定向的信息，因此，自动建模程序的主要任务是避免空间冲突和不可能的构象，而不是优化侧链定向。如果建模的目的是探索功能机制，那么不正确的侧链位置可能会引起误解。

Modeller 是如何建模的

虽然Modeller 整合了序列比对，甚至是序列搜索的工具，但它的出发点是在靶序列和蛋白质模板序列间进行多序列比对。 Modeller 使用模板结构建立一套空间位阻，将它用于靶序列。限制的界限，例如，被建模型中两个残基间的距离，基于模板结构中两个同源残基间的距离。限制也可以用于键角、二面角和成对二面角。通过充分利用这些空间限制，Modeller 可以有效地限制模型可采用的构象数目。

限制的准确形式是建立于同源结构对间的差别的统计分析。这些统计提供的是同源结构间各种性质可能有多大变化的定量描述。例如，对应的α碳到α碳的距离允许变化的数量可以表示为 PDF 格式或概率密度函数。

基于PDF的限制在同源性建模中允许你做的事情是建立一个和模板结构不完全相同的结构。相反地，模型结构允许以一种与已知结构的同源蛋白质之间的差别一致的方式偏离模板。例如，如果模板结构中的一个二面角是 −60度，你使用的 PDF 限制会允许该二面角假设为 60度或者多少有点差距。该值是由在已知同源性结构对中观测到的值决定的，它是依照概率密度函数的形式随机安排的。

基于同源性的空间限制不是应用于模型的惟一限制，还会应用控制合适的立体化学的力场，因此模型结构不能偏离化学原理来满足模板结构的空间限制。应用于模型的所有化学限制和空间限制都组合在函数中（称为客观函数， objective function），该函数在建模过程中会被优化。

ModBase:一个自动产生模型的数据库

Modeller的开发者使被注解的同源性模型的在线可查询数据库变得可用。这些模型使用自动预测管道进行建模。管道的第一步是将每一个未知蛋白质序列和数据库中已存在的蛋白质结构进行比较。然后，和已知蛋白质的结构域有显著同源性的蛋白质可以以这些结构作为模板建立模型。在 Modeller 中使用 *ALIGN2D* 命令把未知蛋白质按照已知蛋白质结构进行比对，使用Modeller程序建立三维结构。管道中的最后一步是评价模型，评价步骤的结果作为搜索结果的一部分提交给 ModBase用户。因为这是专业同源性建模者小组管理的同源性模型开发的所有标准过程，在开始建立自己的同源性模型之前，强烈推荐搜索 ModBase。

使用 Modeller 建立模型的一般过程是，确定未知序列和已知结构的蛋白质的同源性。建立已知结构的多序列比对用作模板，对未知序列进行 Modeller运算。随后，模型可以使用标准的结构评估方法进行评估。

服务器

SWISS-MODEL 是建在瑞士生物信息学研究所（Swiss Institute of Bioinformatics）的一个自动同源性建模Web服务器。SWISS-MODEL 允许提交序列然后会自动返回结构。SWISS-MODEL 使用的自动化过程模拟了同源性建模方案的标准步骤：

1. 使用 BLAST 在蛋白质结构数据库中搜索已知结构的序列
2. 选择模板并寻找基于非同源模板的可建模的结构域
3. 使用建模程序建立模型

   使用分子机制力场优化模型

在 SWISS-MODEL 的 First Approach mode（第一个方法模式）中，必须提供一个未知序列来开始它的运行；然而，也可以选择用于建模过程的模板链。这些信息通过简单的 Web 表单输入。我们可以收到纯 PDB 文件的结果，或者是一个可以使用 SWISS-PDBViewer 打开的方案文件，该 SWISS-PDBViewer是 SWISS-MODEL 服务器的辅助工具，可以下载并安装在自己的机器上。

虽然这看起来很简单，但是这类自动程序很易出错。在非自动分子建模方案中，有许多用户参与的空间。SWISS-MODEL 实际上允许我们使用它们的 Project Mode（方案模式）参与到进程中。在 Project Mode 中，可以使用SWISS-PDBViewer 手动比对模板序列和靶序列，然后写入一个方案文件，并把它上载到 SWISS-MODEL 服务器。

从头开始预测工具

因为序列的从头开始结构预测还没有在很大程度上取得成功，因此我们还不能推荐这方面的例程。如果你对从头开始的结构预测问题很感兴趣，想熟悉该领域里目前的研究情况，我们建议从如下工具中的任何一个开始：Ram Samudrala 开发的 RAMP 软件包，David Baker和 Chris Bystroff 开发的I-Sites/ROSETTA 预测服务

器和John Moult正在进行的工作。最近描述这些方法的杂志文章可以作为进入从头开始的蛋白质结构预测文献的充足的切入点。

把所有的结合在一起：一个蛋白质建模的方案

那么所有这些工具是怎样从序列来建立蛋白质结构模型的呢？本章我们没有仔细描述每一步，因为大多数软件是基于Web的，在这方面都自我解释得非常清楚。然而，你可能依然很想知道如何把它们结合在一起建立一个可靠的建模方案。

作为一个例子，我们为最近的 CASP竞赛 —— CASP 4 中的一个靶序列建立一个模型。我们特意选择了一个困难的序列来建模。但是它在 PDB中没有明确的同源性结构，虽然有线索用来比对靶序列和可能的模板并建模。我们没有声明模型是正确的，它的目的是说明在建立基于距离相似性的蛋白质局部三维模型时可能要经历的过程。

如果有一个明确的、有很强的同源性的已知结构，建立一个起始的同源性模型的过程就会更直接；靶序列和模板简单地沿它们的全长比对，如果必要的话，编辑比对，以一种 Modeller可读的格式输出并提交，或者以 First Approach模式把未知序列提交给 SWISS-MODEL。

寻找同源结构

在任何蛋白质建模方案中，第一步都是寻找基于同源性模型的模板结构（如果可能的话)。

来自 CASP 4 的靶序列 T0101是一种称为 Erwinia chrysanthemi 的细菌的 400个氨基酸的果胶裂解酶 L，我们使用它在PDB 中搜索同源序列。通过使用 PDB SearchFields 表单来启动一个 FASTA 搜索。

返回结果的第一印象是令人沮丧的，因为CASP的目标列表显示在PDB中没有发现和靶序列有很强同源性的序列。没有一个匹配的 E值小于 1，虽然有几个的 E 值小于 10。没有匹配能跨越蛋白质的全长。最长的匹配是有 330 个氨基酸重叠的软骨素酶，其 E值为 3.9。

如图 10-5 所示，得分较高的每一个蛋白质看上去都不相同。最好的匹配是 α-β 桶形结构，然而，第二个匹配（软骨素酶）是点缀着几个 α 螺旋的主要 β 结构，第三个是完全不同的多结构域全 β 结构。

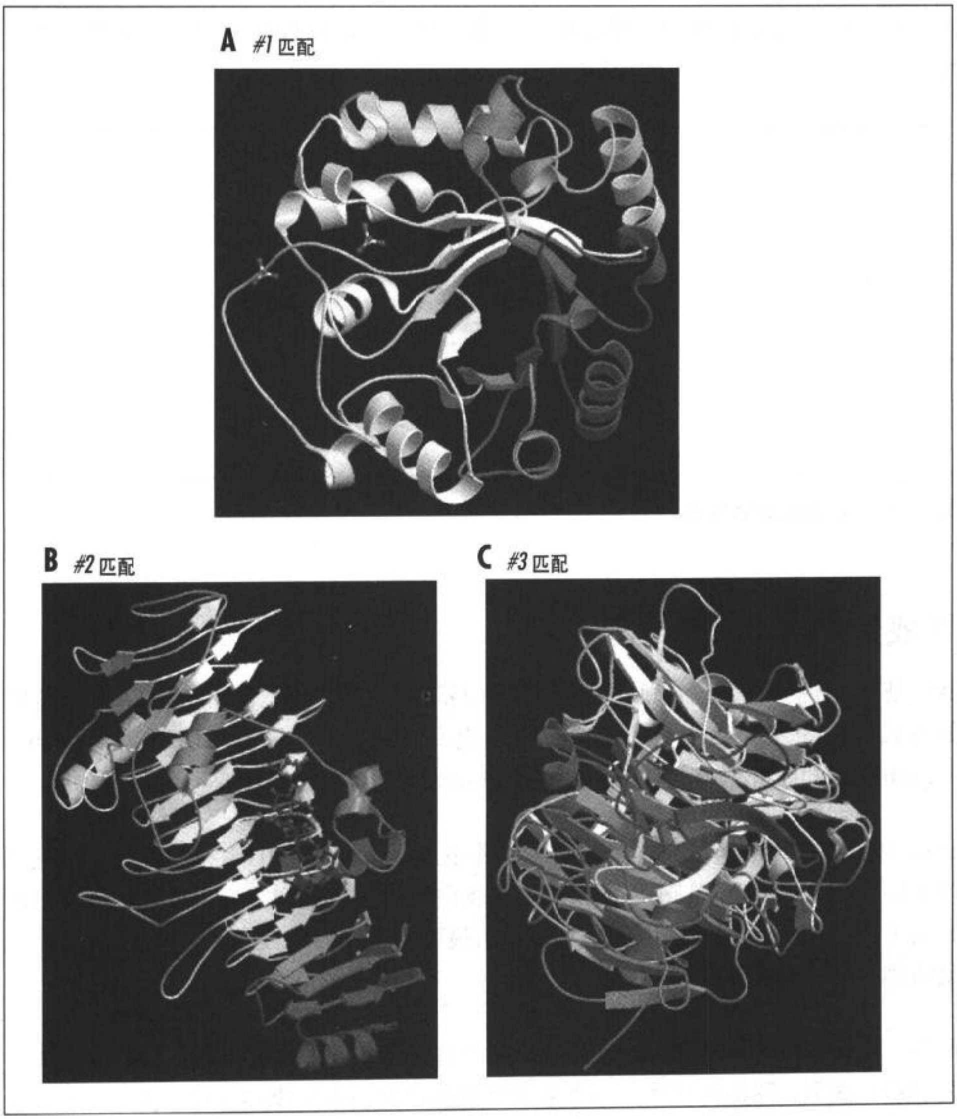


图 10-5　和 CASP 4 中的靶序列相匹配的前三个序列的图像

出于好奇，我们也在PDB中用关键字果胶裂解酶做了简单搜素。列出了8个果胶裂解酶结构，但是很明显没有和靶序列T0101在序列上足够接近可以被认为与序列信息单独相关的。这些结构没有被分为果胶裂解酶L，它们包括果胶裂解酶A、E和C。可是我们发现PDB中的每一个果胶裂解酶分子都有共同的结构特征：被称为β螺线管的巨大的短β链的准螺旋排列，或者更平实地称做单链右手β螺旋（见图10-6）。

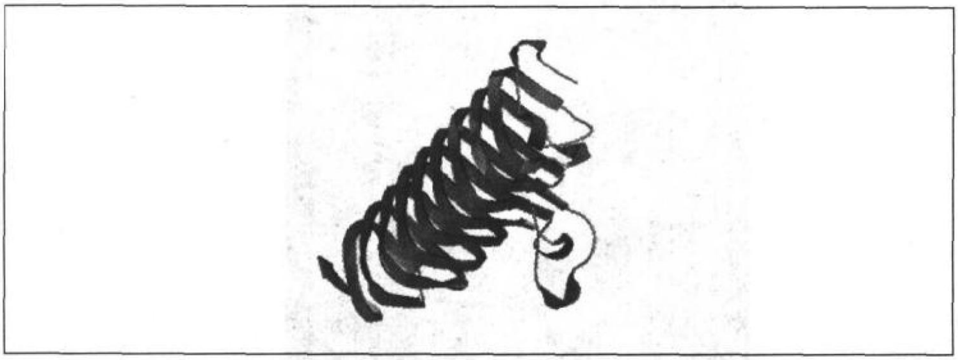


图 10-6　β螺线管结构域

寻找远同源性

我们使用CE检测了已知果胶裂解酶的结构邻居。有趣的是，使用FASTA初次鉴定的和我们的靶序列相匹配的三个蛋白质之一（1DBG，来自 *Flavobacterium heparinium* 的软骨素酶）显示出和许多已知的果胶裂解酶是高得分的结构邻居。

虽然T0101和其他果胶裂解酶的同源性并不显著，但是T0101和它们的结构近邻1DBG间的序列相似性似乎说明我们的靶蛋白质结构可能和其他果胶裂解酶远距离相关(见图10-7)。注意图中的比对显示出很强的保守的结构区域——在分子右侧链轨迹一致的梯状结构。

然而，为了做任何实际的同源性建模，我们需要以某种方式根据潜在的模板结构排列T0101序列。因为没有任何一个果胶裂解酶序列和未知序列有足够的相似性，使我们可以用标准的成对比对来排列它们，因此需要在比对策略上用点技巧。

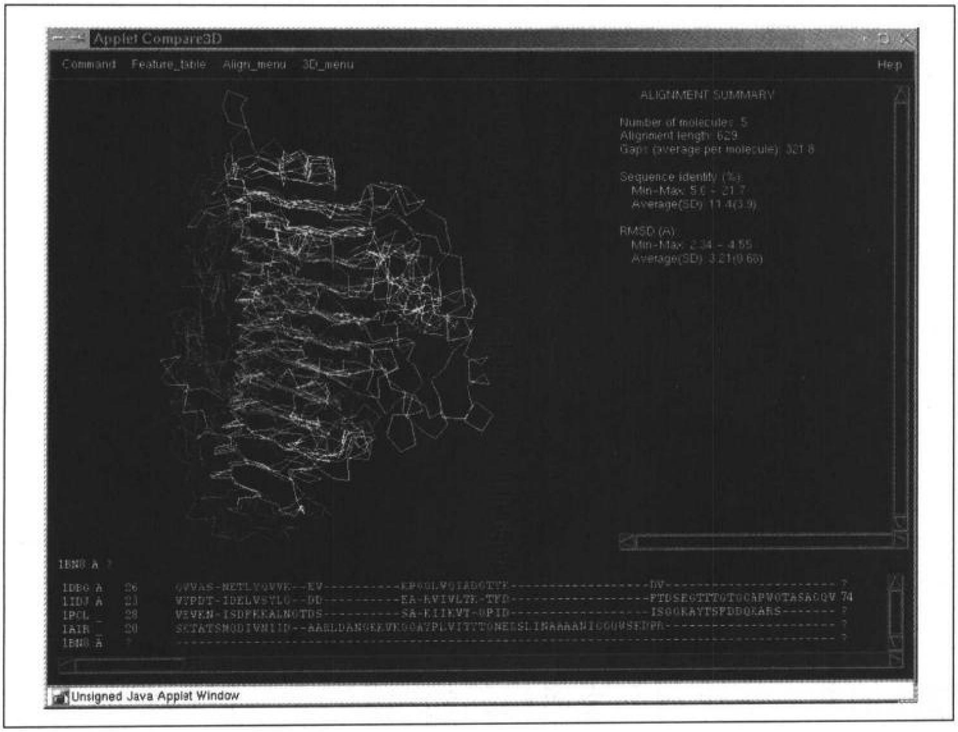


图 10-7　已知果胶裂解酶结构的结构比对 在分子的中心 β 螺线管区域以梯状结构呈现

从序列预测二级结构

利用JPred结构预测服务器对T0101靶序列进行了几种二级结构预测算法的运算。尽管每种方法的预测结果不完全相同，但是我们从图 10-8 可以看出 JPred 的一致性还是很清楚的：T0101 被预测会形成许多短的 β 结构条，这恰恰是形成 β 螺线管所需要的结构模式。

使用穿线法寻找潜在的折叠

我们也将序列发送到 3D-PSSM 和 123D+ 服务器来分析它对各种蛋白质折叠的适合性。来自 3D-PSSM 穿线服务器的得分较高的结果，有 95% 准确性的 E 值范围的包括蛋白质 1AIR（一个果胶裂解酶），1DBG（通过基于序列的搜索已被鉴定为与我们的未知序列同源的软骨素酶），1IDK 和 1PCL，所有的果胶裂解酶都被 CE 鉴定为 1DBG 的结构邻居。这些蛋白质在 123D+ 中也处于结果的上部。

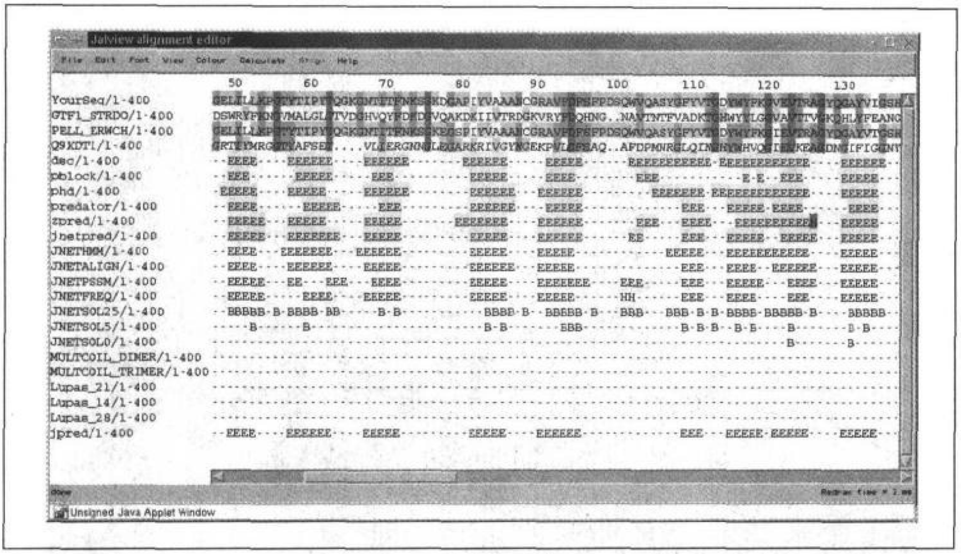


图 10-8　来自 JPred 的 T0101 部分二级结构预测

使用简图方法比对远距离相关的序列

现在我们有多个来源的证据表明结构 1AIR、1DBG 和 1IDK 是我们用来构建未知序列 T0101 模型时所用模板的合理候选者。可是，剩下的挑战是按照模板比对未知序列。我们使用许多不同的方法来进行比对：未知序列与 1DBG 最初的 FASTA 比对；1DBG 和它的结构邻居 1AIR、1DBG 和 1IDK 的 CE 结构比对；在 3D-PSSM 中所提示的每一个数据库中的未知序列与已知序列的预测二级结构的单独比对。当然，由于使用不同的方法，每个比对都不相同。我们选择把未知序列与 1AIR 和 1IDK 的 3D-PSSM 序列到结构的单独比对信息结合成一个比对文件。通过使用 Clustal 的 Profile Alignment （简图比对）模式来相互比对这两个比对而实现这一点。最终，我们以适合 Modeller 的格式把这些比对写入一个文件，将它作为第一个方法的输入。

建立同源性模型

我们为 Modeller 产生如下的输入：

```
The input script,peclyase.top:
```

```
Homology modelling by the MODELLER TOP routine 'model'

INCLUDE                                    # 包含预定义的 TOP例程
SET ALNFILE = 'peclyase.ali'               # 比对文件名
SET KNOWNS  = '1air','1idk'                # 模板代码
SET SEQUENCE = 't0101'                     # 目标代码
SET ATOMS_FILES_DIRECTORY = './templates'# 输入原子文件的目录
SET STARTING_MODEL= 1                      # 第一个模型的索引
SET ENDING MODEL = 3                       # 最后一个模型的索引
                                           # （确定要计算多少个模型）
CALL ROUTINE = 'model'                     # 进行同源性建模
```

我们以 PIR 格式创建一个序列比对文件， *peclyase.ali*，像例子中描述的那样建立并修饰文件来告诉 Modeller 哪一个是靶序列，哪一个是模板序列。

我们也放置了一个 PDB 文件在工作路径的 *templates* 子目录，该文件包括 1AIR和1IDK模板链的结构信息。根据Modeller的要求，文件被命名为 *1air.atm*和 *1idk.atm*，然后运行 Modeller 产生结构模型。模型看上去与它们的模板相似，尤其是在 β 螺线管区，通过包括三维／一维简图和几何评估方法等标准的结构证明方法评价非常好。然而，仅仅像实际的 CASP 4 竞赛那样，我们只能等待实际的 T0101靶结构的发布来验证我们的结构模型。

小结

解析蛋白质的结构是最复杂的，但是像你所看到的那样，有许多软件工具可使它变得更容易。表 10-1 提供了可以得到的最流行的结构预测工具和技术。

表10-1　结构预测工具和技术

目的	原因	可用工具
二级结构预测	作为分类和结构建模的开始	JPred，Predict-Protein
穿线法	检查蛋白质序列和假设已知折叠的适合性；确定远距离相关的结构同源性	3D-PSSM，PhD，123D
同源性建模	根据已知结构的同源性，从序列建立模型	Modeller，SWISS-MODEL
模型确证	检查模型对它的蛋白质序列的适合性	VERIFY-3D，PROCHECK，WHAT IF
从头开始结构建模	在没有同源性时从序列预测三维结构	ROSETTA，RAMP

第十一章

基因组学和蛋白质组学工具

到目前为止我们讨论的方法可以用于分析单个序列或结构，以及比较单基因长度的多个序列。这些方法可以帮助我们理解特定基因的功能或者特定蛋白质的机制。可是，大家可能更感兴趣的是基因的功能是如何表现为生物的可观察性质的：它的表现型（phenotype）。本章我们将讨论一些数据类型和工具，它们正开始用于研究基因组中所有基因的整合功能。

把基因组学和传统实验生物学区分开的是它强调大量信息的自动数据搜集和整合。测验一个基因或蛋白质的实验策略正逐渐被许多基因同时检验的平行策略所代替。我们绝对需要生物信息学来支持平行策略并使得出的数据对生物学界普遍有用。尽管生物信息学运算可能是复杂的，生物信息学和基因组学的最终目的却是直接的。多个来源的信息被整合形成基因组功能及其表达为生物表现型的完整的图画。图11-1显示了从遗传功能移向表现型表达时，我们可能创建的流程图。

从分子水平到细胞水平和更高水平，几十年来生物学家一直在收集这些图片的部分信息。与盲人摸象的故事一样，过于关注生物图片的特殊部分，很容易只见树木、不见森林，给全面观察功能造成了困难。生物化学实验方法的自动化和不断增加，以及把生物化学数据当作公共资源，这一切才刚刚开始。

人类基因组计划不仅会得到数十亿字节的生物序列信息，而且它本身已经开始通过实例改变生物学研究的整个轮廓。蛋白质结构的确定还没有达到序列的确定那样的

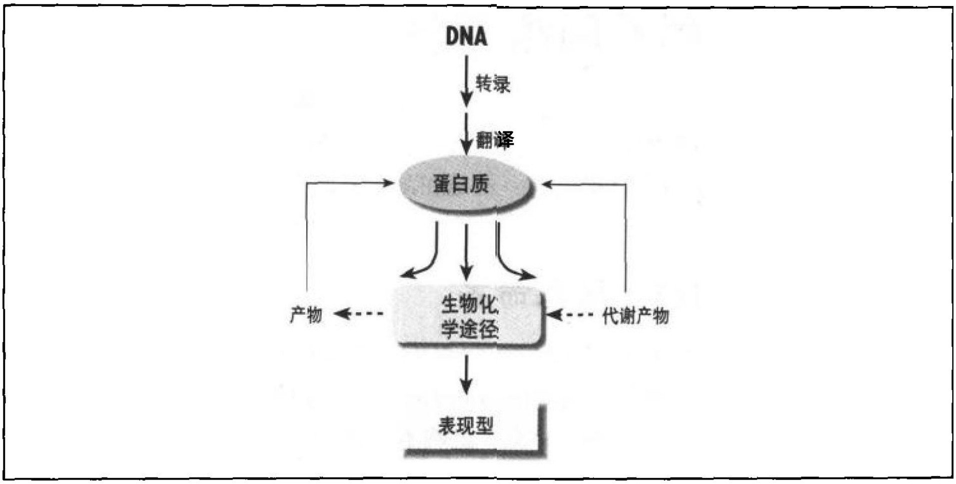


图 11-1　从基因到表现型的流程图

自动化水平，但是结构基因学的几个先行方案正在起步阶段，希望能够产生高速确定结构的管道。在DNA微阵列实验中所包含的概念（成千个精微实验排列在芯片上平行进行）不会一般地转移到其他生化和分子实验类型。尽管如此、所有的生物实验领域都倾向于高效、微型化和自动化。

一长串基因组序列数据的用途本质上和一本没有书眉、页码或者索引的参考书一样。生物信息学的主要任务之一是为信息管理创建软件系统，以有效地用来自功能蛋白质产物的结构（如果有的话）和基因在生物的不同生命阶段的表达效率的一切信息注释基因组序列的每一部分。功能信息是完全注释和理解基因组工作所必需的，目前得到功能信息的惟一途径是传统的生物化学实验，在一个时刻只研究一个基因或蛋白质。不断增长的基因组数据库是未来平行化和自动化生物学研究的动力。

基因组信息管理系统的另一个任务是允许用户在大的数据集之间做直观的、可视的比较。从多个基因组间的可视比较到基因组图谱数据和表达数据的可视化整合的许多新的数据整合方案都正在开发之中。

生物信息学中基因组水平的分析方法的发展明显不如序列和结构分析方法先进。从20 世纪 70 年代开始，序列和结构数据已经可以公开得到；大量完整的基因组只有最近才可以得到。本章我们将集中介绍一些研究团体已经证明在基因组科学中特别有用的数据表示方法和搜索策略。

从基因测序到基因组测序

在第七章中，我们讨论了以某种方式产生大小只差一个碱基的序列片段梯度，通过电泳分开这些片段进行 DNA测序的化学基础。在测定基因（或者基因组）序列的过程中，我们面临的最初的计算挑战之一就是在测序凝胶上解读片段的模式。

原始测序数据分析：碱基命名

从 DNA 测序的原始数据来排列序列的过程称为碱基命名（ basecalling)。作为基因组序列数据的终端用户，我们不能直接访问测序仪输出的原始数据，所以不得不依靠一些处理软件把这些数据指派给序列。虽然不可能真的需要碱基命名软件，但是记住软件的作用和它可能出现的错误还是会有帮助的。

如果这一步不能产生正确的DNA序列，那么随后的任何序列分析都会受到影响。在公共数据库中存放的所有序列都会受到碱基命名错误的影响，这些错误可能是由于测序仪输出的不清晰或者仪器故障引起的。 EST和基因组检测序列的错误率最高（每个碱基的错误率为 1/10~1/100)，然后是小实验室完成的序列（每个碱基的错误率为 1/100~1/1000）和大的基因组测序中心完成的序列（ 1/10000~1/100000）(注 1)。GenBank 的任何序列都可能至少有一个错误。提高测序技术，尤其是 DNA 测序中的信号检测和处理，仍是当前研究的主题。

DNA测序有两个流行的高通量方案。像前面讨论的那样，今天所做的 DNA 测序依赖于产生单碱基分辨率的 DNA 片段梯度和凝胶电泳分离这些 DNA 片段的能力。流行的 Applied Biosystems 测序仪用四种不同的荧光标记物标记 DNA 片段，每一个碱基特异性的片段对应着一个荧光标记物，在同一凝胶泳道里电泳四个样品的混合物。另一个常用的自动测序仪是 Pharmacia ALF 装置，在靠近的单个泳道里，电泳每一个样品。在这两种设备中，凝胶用激光扫描，激光可以激发凝胶上的序列中的每一个荧光带。在四色方案中，荧光信号被一个垂直于凝胶面的激光在一个泳道中同时激发，然后使用四个着色的滤片过滤得到来自每一个荧光标记的不同的信号。在单色方案中，一个平行于凝胶的激光同时从单个测序实验中激发所有四个泳道，荧光被一个射线检测器记录下来。这些方案中的每一个在不同类型的实验中有各自

注 1: 这些值由华盛顿大学的 Sean Eddy 博士提供。

的优点，因此两者都很流行。很明显，硬件的不同导致收集的数据格式的不同，使用数据储存专有的文件格式没有解决这个问题。

有各种商品化和非商品化的自动碱基命名工具。它们有些已完全整合到特殊的测序硬件和输入数据类型中。在确定序列时，它们中的大多数允许，实际上也需要专业用户的校正。

测序的原始结果是测序凝胶上每一位置的荧光强度的记录。图 11-2 显示了一个现代测序实验的检测器输出。自动碱基命名软件的挑战是将明显的荧光峰序列解析为 4 字母的 DNA 序列编码。虽然这看起来很直接，但是对于单个实验中多少序列可读出还是有相当的限制。因为在测序凝胶上带的分离不是很好，分离的质量和带的形状在整个胶长上都会变坏。在测序运行的某些点（通常是在 400 ~ 500碱基处）荧光峰会因变宽和混合而变得不能解析。 DNA 的各种性质引起了测序凝胶的不均一反应，以至于片段的迁移率轻微依赖于片段最后一个碱基的鉴定；整个信号强度依赖于局部序列和实验使用的试剂。当 DNA片段自己折叠起来或者测序引物在 DNA序列的多个位点发生反应引起样品不均一性时，会出现不可读区域。因为这是很好理解的系统性错误，可以开发计算机算法来弥补它们。碱基命名软件开发的最终目的，是通过提供能够解析运行末端更模糊的荧光峰的方法，提高每一个序列读取的准确性，并扩展测序运行的范围。

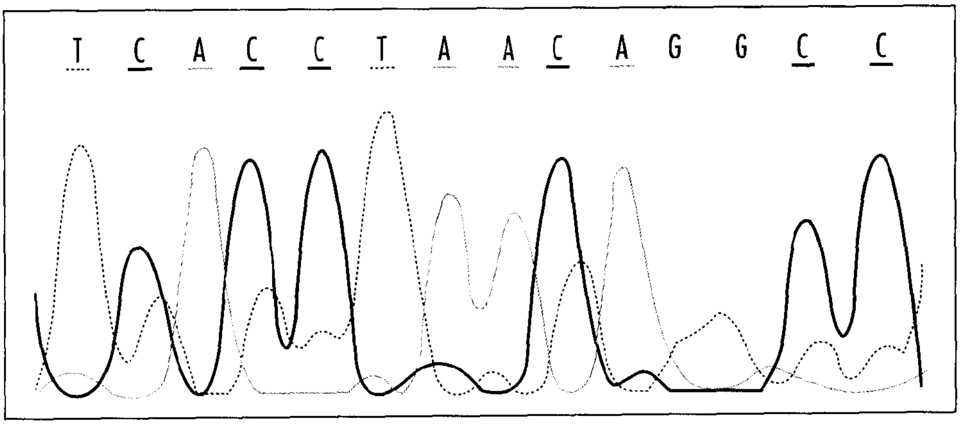


图 11-2　从一个现代测序实验中得到的检测器输出

大多数测序方案通过简单地多次测定基因组的每一个区域和测定 DNA 的两条链(这

可以高质量地读取序列的两端）解决测序过程中的固有错误。如果你读到基因组被4层或10层测序，这意味着基因组的部分已经被多次测序，结果是合并后的最终结果。

现代测序技术采用微毛细管系统代替凝胶，但是方法的核心概念和凝胶测序相同：DNA片段化和电泳分离每个片段。

在这一点上，主要的基因组数据库不对用户提供原始的序列数据，对于大多数应用来说，访问原始测序数据确实没有必要。可是，也许随着计算能力的不断增长，在将来这一点会发生改变，你可能想知道怎样再分析公共数据库中序列的原始序列数据。

Phred是一个碱基命名的非商业软件包，可以从华盛顿大学基因组中心得到。Phred可在Unix或Windows NT工作站上运行。它使用傅里叶分析解析荧光轨迹来预测荧光峰位置的间隔组，之后使用动态规划来使荧光峰的实际位置和预测结果相匹配，然后使用命名错误的概率注释碱基命名的输出结果。Phred得分与碱基命名错误的概率的负对数成比例，因此，Phred得分越高，出现错误的概率越低。这些值在决定基因组区域是否需要重新测序方面是有帮助的。Phred把数据输入一个叫做Phrap的序列组装程序中，然后该程序可以使用序列信息和质量得分来帮助组装序列。

整个基因组的测序

基因组测序不是基因测序的简单的升级版本。像早先提到的，测序运行的序列长度限制大概是500碱基对。基因组长度可从一万到十亿碱基对。因此，为了测定整个基因组的序列，基因组不得不分离成片段，然后测序的片段需要重新组装成一个连续的序列。

基因组测序有两个流行的策略：鸟枪法和克隆contig法。常常结合这些策略来对较大的基因组测序。

鸟枪法

鸟枪法（shotgun）DNA测序是一种完全自动化的DNA测序方法。在鸟枪法测序中，整个基因组或者定义的基因组子集的DNA长度打破为随机的片段。可操作长度的片

段（大约 2000KB ）被克隆进质粒中（所有克隆一起称为克隆库，*clone library*）。质粒（ plasmid）是简单的生物载体，它们可以结合任何随机的 DNA 片段，快速复制产生足够的测序材料。

如果足够大量的基因组 DNA被片段化，一组克隆会多次跨越基因组的每一对碱基。然后，每个克隆 DNA 片段的末端被测序，或者在一些情况下，两端都被测序，这将限制序列装配的方式。虽然只有片段单末端或者双末端的 400~500 个碱基被测序，但是如果从库中随机挑选足够多的克隆进行测序，测序的DNA仍然会多次跨过基因组中的每一对碱基。在鸟枪法测序实验中，通常需要跨过整个基因组6~10次的DNA测序。

鸟枪法测序的最后一步是序列组装，我们将在下一节更详细地讨论。鸟枪法测序实验得到的所有短序列的组装通常不会产生一个单一的完整序列，而是会产生多个 *contig* ——相互不重叠的序列的明确组装的长度。在组装过程中， contig 的开始和结束是因为基因组的区域没有遇到足够的信息（例如，没有足够的代表那个区域的克隆）来继续组合片段。鸟枪法测序测定完整基因组的最后步骤是找到可以填充这些区域的克隆，或者如果在原始库里没有可以填充缝隙的克隆的话，使用 PCR或者其他技术扩增跨越缝隙的 DNA 序列。

最近，Celera Genomics 已经显示鸟枪 DNA测序法（不用映射的测序）可以工作在整个基因组水平，甚至是有很大基因组的生物。果蝇的大基因组序列就是它们成功的证明。

克隆 contig 法

克隆（clone）*contig*法也依赖于鸟枪法测序，但是在更小的规模上。不是从将整个基因组打碎成随机片段开始，该法是从基因组破碎为限制性片段开始，这些片段被克隆进人工染色体载体进行扩增。限制性酶是切断DNA的酶。这些酶是序列专一性的，它们只识别 DNA 中的一种 6~10 碱基长度的特殊序列。在纯粹的统计学中，任何碱基在DNA序列中以 1/4 的概率出现，一个有 N 个残基的序列出现的可能性是 1/4N。切断 DNA序列中的6~10个特异的碱基对的酶以相对稀少的方式切开 DNA，但是因为 DNA 序列不是随机的，导致片段长度特殊模式的限制性酶切是基因组的特征。

每一个克隆的限制性片段可以用标准的鸟枪法测序组装。但是把限制性片段组合成一个完整的基因组是一个不同的问题。当基因组被提取成限制性片段时，它只是被部分提取。用于 DNA样品的限制性酶数量只能够切开样品中大约 50%的酶切位点。这就意味着有些片段会跨过某个特殊的限制性位点，而另一些片段会在那个特定位点切开，而跨过其他的限制性位点。因此，这些限制性片段组成的克隆库会包含重叠片段。

染色体步行(walking）是开始于一个特殊克隆，然后寻找和它重叠的另一个克隆，之后是再下一个克隆的进程。探针杂交或者PCR方法可用于鉴别插入每一个克隆的限制性片段。有许多实验策略可以用这种方法少花费时间而建立起基因组图谱。基因组图谱是对基因组中已知特征的位置的记录，它使得通过探针杂交或者其他方法把特定的克隆序列和基因组中专一的位置结合起来变得相对简单。

基因组可在各种详细程度上作图，遗传学家过去常常根据遗传连锁图谱思考，遗传连锁图谱大致上把显现出特殊性质的基因安排到染色体特殊的位点上。然而，遗传连锁图既不能提供足够的细节来支持整个基因组序列的组装，也不能指出对应着某个特定性质的实际的 DNA序列。尽管，基因连锁图提供的是一套有序的标记，有时其细节依赖于生物，这些标记可以帮助研究者理解基因的功能（并提供组装全基因组图谱的框架)。

物理图谱可用几种方法建立：通过切割特殊位点的限制性酶提取DNA；通过开发有序的克隆库；最近通过固定在玻璃底物上的单一限制性酶切割的 DNA分子的荧光显微方法。每一种方法的关键是结合使用标记探针和已知遗传标记（在限制性图谱中）或通过鉴别重叠区（在库建立中），来将基因组片段正确排序为高度专一的图谱（见图 11-2)。

LIMS：追踪所有小序列

在执行测序方案时，追踪从基因组中分离出的成百万个独特的DNA样品是信息技术最大的挑战之一。从科学角度出发，它也许是最不令人激动的事情之一，因为它涉及到追踪每一个样品来自于基因组的位置，哪一个样品进哪一个容器，每一个容器可能进入的位置——一个巨大的样品储藏系统，最后哪一个数据来自哪一个样品。处理来自高通量的测序输出的系统称为LIMS(Laboratory Information Management Systems，实验室信息管理系统）。尽管我们在本书不会讨论它们，但是 LIMS 的开

发和维护组成了工业设置中生物信息学工作的最大的一份。其他的高通量技术，例如微阵列和化学信息学，也需要复杂的 LIMS 的支持。

序列组装

碱基命名只是把完整的基因组序列结合到一起的第一步，一旦得到序列短片段，这些序列必须要组装成可能是它们长度的数千倍的完整序列。下一步是序列组装。

除非刚好在为一个基因组计划工作，否则序列组装不是你自己可能要大规模去做的事情。然而，即使是小实验室也可能要解决一些需计算机协助的序列组装问题。

使用鸟枪法的 DNA 测序提供了成千上百万个小序列，每一个片段长度有 400~500 个碱基对。这些片段是随机的，相互之间可以部分或者完全重合。因为这些重合，组中的每一个片段通过序列鉴别，可以鉴定为和其他一些片段相邻。这些片段中的每一个也和另一组片段组重合，依此类推。这是对一些片段两端已知的序列和只有片段一端已知的序列使用的标准方法。图 11-3 显示了鸟枪测序的方法。

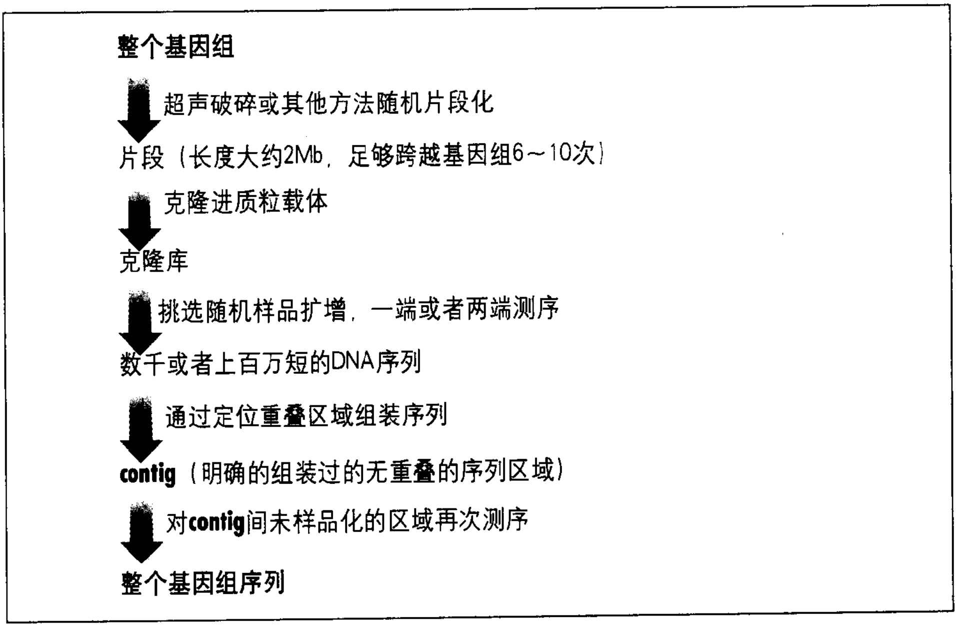


图 11-3　鸟枪 DNA 测序方法

最终，所有的片段需要恰当地连成一个不间断的序列。片段间序列重叠的鉴定对序列如何组装设置了一些限制。对一些片段来说，序列长度和序列的两端是已知的，这给序列如何组装设置了更多的限制。组装运算努力满足所有的限制，对组成基因组的所有片段产生一个最适的顺序。

重复序列可能使组装过程复杂化。一些片段不能克隆，测序过程会失败，在 DNA序列中留下的空隙必须通过再次测序解决。这些空隙使得自动组装复杂化。如果在序列的一些点上没有足够的信息继续组装，正在建立的序列的 contig就会中断，当有足够的序列信息可以使组装继续的时候，一个新的 contig 重新开始。

华盛顿大学基因组中心的 Phrap程序可以有效地组装序列片段，虽然这需要大量的计算时间。程序 Consed是 Unix下的Phrap序列组装输出的编辑器。TIGR Assembler是另一个基因组组装软件包，它对小基因组、BAC 或 EST 工作良好。

在 Web 上访问基因组信息

来自几百个基因组的部分或完整的 DNA序列可以在GenBank 中得到。把这些序列记录整理成可理解的基因组结构表示法并不容易。现在正在努力以一种用户友好的格式把DNA序列和更高层次的基因组图谱整合到一起。到目前为止，这些努力着眼于人类基因组和一些重要的动植物模式系统的基因组。目前，它们还没有被集合成一个统一的资源，虽然 NCBI的目的的确是广泛地覆盖可得到的基因组数据。

看一个基因组数据就像看一张世界地图一样。你可能想从远处看地图，想看到大陆和国家的整个布局，或者你想知道从 A点到 B 点的捷径。每一个选择需要不同类型的地图。然而这些地图需要联系在一起，因为你可能希望知道从圣地亚哥到布莱克斯堡、VA的一般方向，也可能想确切地知道到那里以后，如何找到 Virginia Tech校园里的一个特殊的建筑。基于 Web 的基因组分析工具所采用的方法与 MapQuest这样的在线地图数据库所采用的方法相似。地名和邮政编码就像基因的名字和GenBank ID号。可以按照你的愿望进行专门的搜索，或者从基因组的全图或者放大图开始。

基因组图谱资源也有和在线地图资源一样的限制。可以搜索 MapQuest，查找布莱克斯堡的每一条街道，但是如果问 MapQuest在 Cochin 和印度西南海岸的Mangalore

的乡间小路的捷径，它就不能帮助你了。目前， NCBI和EMBL提供了查看人类基因组的详细的地图和工具，但是如果你的主要兴趣是猫的基因组，那你就很不幸了(至少今年是这样的)。

基因组资源也受限于生物信息学分析方法的能力。在基因组站点可以得到的分析工具通常局限于序列比较工具，为基因组提供的任何单序列特征检测工具和可以从其他数据库无缝整合进来的关于该基因组的信息。如果希望在基因组站点使用工具做点事情，你不能使用现存的序列和结构工具，你仍然会很失望。基因组站点的确提供的是高度自动化的用户经历和相关概念及条目的专业辅助连接。这是一个有价值的优点，但还是有一些问题不能回答。

NCBI 基因组资源

NCBI在其主站点的 Genomic Biology （基因组生物学）部分为访问基于 Web 的基因组分析工具提供了广泛的选择。这些工具是为寻找专业问题答案的生物学家而设计的。不需要基本 Web 技巧和生物学知识以外的东西就可以对感兴趣的问题使用这些软件。它们的界面完全是指向并点击（point-and-click）的，NCBI 提供了帮助学习怎样使用这些工具和数据库的丰富的文档。

下面是可得到的工具列表：

基因组信息

基因组计划的信息可以从 NCBI 的 Entrez Genomes 页面得到。数据库列表可以是整个数据库的，也可以是相关生物群的，如微生物、古细菌、真细菌、真核和病毒等。数据库的每一个条目链接到分类浏览页的条目或者有进一步到该生物其他信息的主页的链接。如果生物的基因组图谱可以得到，在该生物的主页上会显示一个“ See the Genome ”的链接。从主页上也可以下载基因组序列和参考文献。

图谱阅览器

如果可以得到生物的基因组图谱，可以在首先显示的卡通图上的某些部分点击，访问几个不同的查看选项。依赖于基因组，可以访问概貌图的链接、显示已知蛋白质编码区域的图谱、蛋白质和 RNA 编码区域的列表及其他信息。信息通常以文本格式下载。图谱浏览器（Map Viewer）可以区分四种水平的信

息：生物主页、基因组图形查看、每条染色体的详细图谱（用户可以从主图选择放大）和序列查看，它用图形显示对基因组序列区域的注释。在写作本书时，图谱查看器的完整功能，还仅仅用于人类和果蝇的基因组；然而，对于任何完整的基因组，可以观察到可点击的基因组图谱和在序列水平已注释的基因组。

ORF 寻找器

ORF（Open Reading Frame，开放阅读框）寻找器（Finder）是一个在 DNA 序列中定位阅读框的工具。ORF寻找器使用标准的或者用户指定的遗传密码翻译序列。在未编码的 DNA 中，频繁地发现终止密码子。只有没有终止密码子的长的不间断的序列才可能是编码区。ORF 寻找器可以提供关于 DNA 序列正确阅读框和编码区起始和终止点的线索。对于 Entrez Genomes 数据库中的许多基因组，ORF 寻找器可以作为基因组图谱查看的整合工具。

LocusLink

LocusLink是几个真核生物中已知表现型的遗传定位的辅助数据库。该数据库提供了性质的字母列表以及到 HomoloGene 和 Map Viewer 的链接。

HomoloGene

HomoloGene 是一个成对直向同源（因物种形成而分离的不同生物间的同源基因，与因基因重复而分离的共生同源相对）的数据库。包括四个主要的真核基因组：人类、小鼠、大鼠和斑马鱼。直向同源对或者通过文献报告的校正或者通过相似性计算而被鉴定。HomoloGene 可以通过基因符号、基因名称、GenBank 访问号和其他特征进行检索。

COG（Cluster of Orthologous Group）

COG是一个直向同源蛋白质群的数据库。数据库是通过比较 21 个完整基因组的蛋白质序列而开发出来的。COG 中的条目代表在进化历史上大多数时候都是保守的基因组功能 —— 那些发育很早，在所有已知的完整基因组中都保留的功能。作者的假设是这些原始的保守序列包含着现在的物种（即，已生存到基因组测序时代的物种）所需要的最小功能核心。COG 数据库可以通过功能分类、系统发育模式和许多其他性质来检索。

NCBI 也提供了几种重要的真核生物基因组的与基因组有关的资源的详情，包括人类、果蝇、小鼠、大鼠和斑马鱼。

TIGR 基因组资源

TIGR（Institure for Genome Research，美国基因组研究所，*http://www.tigr.org*），与其他一些重要的人类基因组测序中心和Celera这样的商业公司一样，是新基因组序列数据的主要生产者之一。TIGR的主要测序计划是微生物和农作物基因组，以及人类的第 16条染色体。TIGR 最近宣布了 Comprehensive Microbial Resource(综合微生物资源)，包括他们已测序的所有基因组的完整的微生物基因组。目前，每一个微生物基因组都有自己的网页，从那里可以查看各种基因组。在资源内也有允许搜索 *omniome* 的工具，因为 TIGR 在它的数据库里设计了组合的基因组信息。 TIGR 的工具不像 NCBI 基因组分析工具那样可视化。选择检测区域需要在表单中输入特殊的信息，而不仅仅是在基因组图谱上指向和点击。然而， TIGR 资源是 NCBI工具的有用补充，它对相同的遗传信息提供了不同观点。

TIGR维护着许多基因组专一性数据库，这些数据库着眼于表达序列标签（EST)，而不是完整的基因组数据。EST是来自于 cDNA 克隆的任何一端的部分序列。尽管它是不完整的，EST 对于实验分子生物学家仍是有用的。因为 cDNA库通过产生细胞 mRNA（信使 RNA）的互补 DNA 而建立，cDNA 库对在特定组织或细胞内的实际基因的表达情况提供了线索。因此，和一个 EST 相匹配的序列可能是帮助鉴定新基因功能的起始步骤。 TIGR 的 EST数据库可以使用简单的基于 Web的界面形式通过序列、EST 鉴定者、cDNA 库名称、组织或者基因产物名称来检索。

EnsEMBL

EnsEMBL 是 EMBL、EBI和 Sanger 中心（*http://www.sanger.ac.uk*）的合作计划。它可以自动跟踪人类基因组的测序片段，把它们组装成一个长链。然后，可对组装的序列使用基因寻找和性质寻找工具、序列比较工具等自动分析方法，用户可以通过 Web 界面使用它们。

2000年 6月人类基因组协会宣布完成了人类基因组草图。需要强调的很重要的一点是，这些图谱和现在实际已得到的基因组信息的大部分，只是最终图谱的一个草图，最终图谱还可能要花许多年来完成。为了保持有用，草图必须不断更新以和不断更新的序列数据库同步。从人类和小鼠的基因组开始，EnsEMBL 计划希望将它的自动数据分析管道用于许多不同的基因组。

有三种方式可用来搜索 EnsEMBL：用查询序列对该数据库进行 BLAST搜索；用已知的基因、转录或图谱标记 ID搜索；或者是染色体图谱浏览器，它允许你挑选染色体，并可放大成更专一的区域。所有这些工具都有相当好的自我解释性，可从 EnsEMBL 的 Web 网站得到。然而，为了使用它们，应该知道一些你要寻找的东西的知识和要查看的是哪条染色体。

其他测序中心

TIGR不是惟一为基因组数据分析提供软件和在线工具的基因组中心。基因组测序程序通常和生物信息学成分结合在一起，吸引对开发生物信息学方法感兴趣的研究者；他们的 Web 站点是进入生物信息学世界很好的入口。华盛顿大学基因组中心因开发广泛使用的 Phred和 Phrap 序列组装工具而闻名。其他基因组中心包括 Sanger 中心、DOE联合基因组研究所（DOE Joint Genome Institute）、St. Louis 的华盛顿大学和许多公司的中心但不仅限于这些。

进行中的基因组测序计划和活跃的基因组中心的完整列表可以从 GOLD（Genomes OnLine Database，基因组在线数据库）在线得到，它是由 Integrated Genomics 公司（*http://wit.integratedgenomics.com/GOLD/*）维护的公共站点。

生物专一性资源

TAIR（The Arabidopsis Information Resource，拟南芥信息资源）是生物专一性基因组资源的优秀代表，它着眼于广泛使用的植物模型系统拟南芥（Arabidopsis thaliana)。除了 EnsEMBL和 NCBI提供的标准属性，如可点击和放大的染色体图谱及序列分析工具外，TAIR 还为资源用户提供了各种其他信息的专业辅助链接。TAIR 限定在拟南芥范围内，但它是比一般的公共数据库更深入的资源。从玉米到斑马鱼的许多生物都有相似的资源。在线基因组资源的列表可能位于 GOLD、NCBI 和 EMBL 等几个站点。

注释和分析整个基因组序列

在序列的存储和分析上，基因组数据呈现出全新的问题：

基因组序列非常大。

用户需要在整体上和在有意义的片段上访问基因组数据库。

基因组中序列的大部分并没有相应的已知功能。

可以通过几种方法实现对带功能信息的基因组的注释：和序列数据库中已存在的生物信息进行比较，和基本文献中已出版的信息进行比较，使用ORF检测和基因寻找的计算方法，或者通过基于序列比较的进化关系推理实现从一个基因到另一个基因的信息传播。由于可以得到全部数量的数据，自动注释是可以实现的，但是必须不会自动传播错误。使用计算方法十分易犯错误，序列相似性搜索可能会导致没有生物学意义的提示，genefinder常常在准确检测基因的起始和结束时有困难。有时，实验信息是不对的，或者在数据库中被错误地记录。使用这个信息去注释基因组会在数据库中留下错误的残余，然后使用比较的方法，错误可能会进一步扩散。

基因组注释

基因组注释是一项困难的工作，一部分原因是由于附加到基因组每一个基因上的非常多的不同的信息条。对每个用户来说，不是对每一条信息都有兴趣，也不是每一条信息都可以（或者应该）塞进基因的单个文件中。基因组的注释依赖于关系数据库对基因组序列和其他数据信息的整合（注 2）。

基因功能最初的信息来源是实验室的实验。需要做很多实验来指出一个基因的功能。理想状况下，所有的不同实验数据都在某种程度上与基因的注释有关，这实际上意味着在多个数据库之间的内容的超链接 —— 序列、结构和功能基因组完全链接在一起的查询系统。虽然“世界范围的数据库”（在用户的感觉上）的目标还没有达到，也许永远也不会达到，但是这个策略已经开始被大多数主要的公共数据库所采用。

MAGPIE

MAGPIE是一个基于序列相似性的基因组注释的环境。它可以保持基因组计划的状

注 2：术语“关系数据库（relational database）”会给你一个线索：数据库的功能是保持条目间的关系或连接。我们将在第十三章更详细地讨论。

态信息，使得可以在 Web上得到基因组的信息，同时提供基于自动序列相似性的和手工的注释界面。即使没有维护一个公共使用的数据库，看一下 MAGPIE 的特征也可能会帮助你明白一些关于基因组注释的信息技术问题。*Sulfolobus solfataricus* P2 基因组计划和其他许多更小的基因组计划已经在使用 MAGPIE；*S. solfataricus* 计划在它的网站上提供 MAGPIE 在基因组注释过程中所扮演角色的信息。

基因组比较

两个和多个基因组的比较是对许多研究都有用的思想，包括进化生物学的基本问题到非常专业的临床问题，例如遗传多态性的鉴定，它会导致疾病加重或者是表现型的重大变化。

为什么同时比较整个基因组要好于仅仅比较基因呢？随着人类基因组计划的完成，研究者正开始详细探索基因组结构是如何影响基因组功能的。垃圾 DNA 真的是垃圾吗？是DNA的结构特征在控制表达吗？有我们还没有找出的启动子和调节区域吗？基因组比较可以通过指出未定性或者甚至是已假设为多余DNA 中的相似性区域来回答这些问题。基因组比较也会帮助基因组注释。表现型基因组比较已经在帮助校正其他基因组的测序；人类基因组和小鼠基因组之间的比较就是一个这样的例子。基因组比较在图谱和序列水平上都是有用的。

PipMaker

PipMaker 是每个 DNA序列高达 2MB 的两序列之间的比较工具（更长的序列需用新的 2.0 版本，很快就会发布），它会产生一个相同百分比的散点图作为输出。这在鉴定更长序列的大规模模式的相似性时很有用，虽然明显不能用于完整的更大的基因组。PipMaker 的使用方法相对简单。从两个 FASTA格式的序列文件开始，为了隐藏序列重复，首先产生一套指令（用 RepeatMasker 服务器产生），这会减少序列比较中的没有信息的打击的次数。产生的信息结果，加上一个包含已知基因位置数字列表的简单文件，提交到 Penn 州立大学的 PipMaker Web服务器，结果会发邮件给你。在网站 *http://bio.cse.psu.edu/pipmaker/* 上可以得到 PipMaker 的详细指导手册。PipMaker 依赖于 BLASTZ 比对序列。BLASTZ是由 NCBI 开发的 BLAST测试版本，设计用于极长序列的比对。

MUMmer

基因组成对比较的另一个程序是 TIGR 的 MUMmer。MUMmer的设计目的是迎合 TIGR 的测序计划的需要，它对预计比较相似的微生物的基因组比较进行了优化。它的第一次应用是详细比较了 *M.tuberculosis* 的两个菌株的基因组。MUMmer可以比较数百万个碱基长度的序列，产生相似区域的彩色可视效果。MUMmer 基于一种叫做 *suffix tree* 的计算机算法，该算法可使系统很容易快速处理大量的成对比较。用于标准的 BLAST 比较的动态规划算法不能很好地衡量序列长度。对于基因组长度的序列，动态规划方法慢得让人不能忍受。

MUMmer是为了解决使用标准的序列成对比较进行全基因组序列比较的问题而开发的新方法之一。MUMmer是为 Unix 系统设计的，非盈利的研究所可以免费得到。在 TIGR 网站上有 MUMmer 的一种新的公共 Web 界面。

功能基因组学：新的数据分析挑战

高速测序方法的出现改变了我们研究编码蛋白质的DNA序列的方法。我们一度把找到生物基因组中的这些 DNA作为目标，而不必考虑太多的上下文关系。现在作为一个单一实体查看染色体的整个DNA序列成为可能，并可以检查它的各部分如何一起工作来产生整个生物的复杂性。

基因组功能可以松散地分为明显的几类：代谢、调节、信号和结构。代谢途径把来自环境的化学能（如食物）转变成对细胞有用的东西。调节途径是控制基因组DNA做什么，什么时候表达，什么时候不表达的生化机制。基因组调节不仅包括表达的基因，而且包括调节蛋白质可能结合的 DNA结构和序列信号。其中信号途径控制细胞中从一个部分到另一个部分的化学物质的移动。梳理组成这些途径的相互作用网是生化学家和分子生物学家的工作。许多控制 DNA转录的调节系统已在开发中。将这些代谢、调节和信号系统绘入基因组序列是功能基因组学领域的目的。

基于序列的基因表达分析方法

除了基因组序列之外，GenBank 中还包括许多其他种类的 DNA序列。生物的 EST（expressed sequence tag，表达序列标签）可能是基因表达定位探索的非常有用的

起点。为了理解为什么，需要回忆一下 EST 代表什么。EST是 cDNA克隆的部分序列；cDNA 克隆是以细胞 mRNA 为模板合成的 DNA 链（注 3）。在细胞中，mRNA是带有信使的 RNA ——不久可以转变成蛋白质。mRNA 的水平反映了细胞或者它的环境的变化；mRNA水平是组织依赖性的，它们也随着生物的生命周期而变化。mRNA或者 cDNA 的定量提供了在特殊条件下基因组正在做什么的很好的测量方法。

来源于 mRNA模板的 cDNA分子序列应该和作为mRNA模板的原始的DNA 序列相同。测序一小段 cDNA 序列提供了定位基因组序列中 mRNA 来源的足够信息。

NCBI提供了一个叫做 dbEST 的数据库，该库可访问几千个 EST库。相当大的一部分是人类 EST库，但是也有几十个其他生物库。NCBI的 UniGene数据库提供了对人类、小鼠、大鼠和斑马鱼的专一性 EST 库的完全可搜索访问。UniGene 的 EST数据库和已定性的基因序列相结合并聚类，使用自动的聚类程序来鉴定相关序列的集团。Library Browser （库浏览器）可在 UniGene 中定位感兴趣的库。

基于序列表达分析的另一个 NCBI 资源是 SAGEmap。SAGE（Serial Analysis of Gene Expression， 基因表达系列分析）是一项实验技术，使用它许多基因的转录产物可以通过在序列中特殊位置（通常在限制性位点）的 DNA短标签测序快速定量。SAGEmap是对 SAGE 用户群的特殊的在线资源，它可以鉴定人类 DNA序列中潜在的 SAGE标签，并把它们绘制到基因组中。

DNA 微阵列：功能基因组中的新兴技术

最近，新技术使研究者快速探索整个基因组的 DNA 表达模式成为可能。微阵列（microarray， 或者基因芯片）是一厘米见方的小玻璃片 ——像显微玻片。在玻璃片的表面覆盖着20000或者更多个精确放置的点，每一个点包括不同的 DNA寡聚物(短多聚核苷酸链)。cDNA 也可以结合到玻璃片上作为探针。其他介质，如薄膜也可用来代替玻璃片。实验的关键是每个DNA 片段都是固定的，一端结合到玻璃片的表面。引起微阵列信号改变的任何反应可以精确地安排给一个专一的 DNA序列。

注 3： 术语“ transcriptome ”用来描述特定生物的已测序转录本的集合。

微阵列实验提供了 DNA 重要性质的资本。DNA（或者RNA）用一条链可以和互补的 DNA 杂交。如果两条链的互补是完美的，两条链的结合就不容易打开。 DNA 微阵列中的每一个寡聚物可作为探针检测惟一的互补DNA或RNA分子。这些寡聚物可以被荧光标记的DNA结合，芯片允许用共聚焦或CCD相机扫描。运行于芯片上的DNA样品的互补序列的存在与否决定了阵列该位置是否会亮起来。因此，平均20000个序列的存在与否可在一个基因芯片上用实验证明。

微阵列与传统的杂交实验如 Southern Blot （用标记的探针序列探测滤膜上分离的DNA样品）或者 Northern Blot （探测滤膜上分离的RNA 样品）没有概念上的不同。在传统的印迹实验中，蛋白质样品是固定的，在微阵列中，探针是固定的。一个微阵列实验所收集的信息数量比传统印迹实验要大得多。图11-4仅仅显示了拟南芥微阵列扫描的一部分[图像由AFGC（Arabidopsis Functional Genomics Consortium，拟南芥功能基因组协会）和 Stanford Microarry Database（斯坦福微阵列数据库，*http://genome-www.stanford.edu/microarrays*）提供]。另一个优点是微阵列实验依靠荧光探针而不是杂交技术中使用的放射性探针，基因芯片可以由机器制造，而不是通过艰苦的手工劳动。

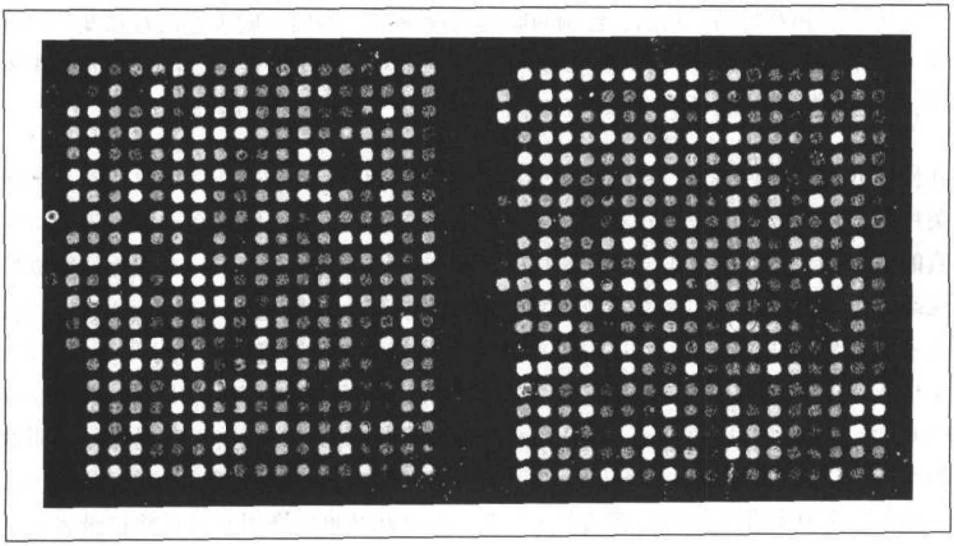


图11-4　微阵列扫描

微阵列技术现在常用于 DNA测序实验；例如，在检测多态性的存在时。最近的另一个发展是用于基因表达分析。表达基因时， mRNA 的转录体就会产生。如果和感兴趣基因互补的 DNA寡聚物位于微阵列中，mRNA或者 cDNA就可以杂交到芯片上，从而对基因表达与否做出快速的检验。这样的实验已在酵母中进行，测试环境糖浓度变化会引起整个基因组表达模式的不同。微阵列实验可以提供生物的每一个基因对环境改变的反应的行为信息。

微阵列设计和分析的生物信息学挑战

那么，为什么微阵列会在生物信息学的书中占有一节呢？因为生物信息学在微阵列实验中扮演着多种角色。实际上，没有计算机和数据库的参与，要想微阵列有用是很困难的。从设计为了特殊目的的芯片，到信号定量，再到链接表达简图的基因群的获得，靠眼睛和纸笔来做，即使不是不可能的话，微阵列分析也是一个十分困难的过程。

在写作本书时，最流行的微阵列实验的实验室设备是 Affymetrix机器 ;然而，自制装置和它的差距已经不远了。如果用的是商品化的阵列仪，整合性的软件可能使数据分析变得相对容易。然而，自制的微阵列设备可以输出不同大小的数据集。阵列仪可能不如商品化机器点得均一。标准化是很困难的。使用自制设备意味着不得不寻找支持微阵列实验所有步骤并且有需要的数据分析特征的软件。

在使用非商品化设备进行微阵列实验时遇到的主要挑战之一，是链接表达数据和相关序列与注释的工具数量很有限。对一个新手来说，构建这样一个数据库界面可能真的是一种负担。一些基于专有芯片格式的专有软件，经常对芯片专一性的数据库尾端有很好的支持，但是对各种图像格式和数据集大小，它并不总是可以推广的，或者是容易这样做的。在公共领域，改善学术研究者这种处境的几个计划正在进行，包括 NCGR的 GeneX和EMBL-EBI 的 ArrayExpress。NHGRI（National Human Genome Research Institute ，美国国家人类基因组研究所）目前正在提供一个叫做 ArrayDB（*http://genome.nhgri.nih.gov/arraydb/*）的阵列数据处理系统的测试版本，包括分析工具和相关的数据库支持（注 4）。ArrayDB也允许用户社区将记录添加到中心数据库中。

注 4:　在写作本书时，发行版为 α测试版。

斯坦福的 Pat Brown 小组在他们的网站上列出了详尽的微阵列资源名单，包括建立自己的阵列仪的指导（只花费商品化设备的 10% 左右）和控制阵列打印的 ArrayMaker 软件。这个网站是微阵列初学者的优秀资源。

计划阵列实验

微阵列实验的一个关键要素是芯片设计。这是使用商业设备和芯片的用户常常忽略的一个方面，因为这些系统的好处之一就是，甚至是在考虑做实验之前，专家已经为你设计好了芯片。芯片设计的过程可能要花好几个月的时间。

即使是最大的芯片也不能适合真核基因组中的所有蛋白质；那可能有数百数千个不同的靶点。芯片设计者不得不选择基因组的子集，当它们一起被检测的时候，可能提供有用的信息。EST数据集在设计微阵列引物时可能会有帮助；虽然它们没有定量信息，EST 的确显示了在特定环境下哪些基因可能是活跃的，因此对特定的微阵列实验有信息价值。

为了使微阵列实验的结果清晰明确，阵列中的每一个 DNA 探针必须足够单一，使得只有一个特殊的靶基因可以和它杂交。否则，在每一个点检测到的信号数量将不能正确定量。

实际上这意味着许多序列分析：寻找可能感兴趣的基因、选择并合成合适的探针。一旦序列被选择了，它们的序列，加上阵列中每一点的可能的背景信息，必须记录到一个数据库中，使得当分析结果时，信息是可以访问的。最后，数据库必须足够健壮，以能够考虑到公共数据库的注释和信息的改变从而避免对结果进行错误注释。

一些探针和引物设计的资源可以在 Web上得到。一个“好的”寡核苷酸 ——可以用作微阵列、PCR和其他杂交试验的探针或者引物，应该不能自身杂交形成二聚体或者发夹结构。它应该惟一地和你感兴趣的靶序列杂交。对于 PCR应用，引物必须有合适的熔解温度和稳定性。一个优秀的设计 PCR 引物的网络资源是 Whitehead研究所的 Primer3 网站；CODEHOP 是另一个引物和探针设计应用程序，它是基于 Blocks 数据库中发现的序列基序的。

用 CrazyQuant分析扫描的微阵列图像

一旦微阵列实验结束，你会发现自己拥有许多包含微阵列扫描图像的很大的TIFF文件。如果没有使用整合的分析工具包，该怎么办呢？

公共域微阵列分析工具的标准是斯坦福大学开发的工具包。一个叫 ScanAlyze 的工具包是这组程序中用于 Windows 的图像分析工具。ScanAlyze 被认为很好并被广泛使用，尤其是在学术界。它以半自动化的栅格定义和多通道像素分析为特征，并支持 TIFF 文件和斯坦福 SCN 格式。到目前为止它是这儿讨论过的最复杂的图像分析程序。

一个相对直接的公共域阵列图像分析程序是 CrazyQuant,它是来自于华盛顿大学的 Hood 实验室的 Java 应用程序。CrazyQuant 是菜单驱动的，可以使用 TIFF、JPG 或者 GIF格式的微阵列图像。CrazyQuant 假设了一个有规律间隔点的二维介质。为了开始分析，必须定义精确指出点位置的二维栅格。然后程序计算每一个点的相对强度并把它们存储为整数值。 CrazyQuant 可定量一色和两色（不同的）的阵列数据。CrazyQuant 在 Linux 工作站上非常容易安装。下载压缩包，把它移到自己的目录中，解压缩，通过输入 *java CrazyQuant* 可运行程序。压缩包中有一个简单的 GIF 图像，可帮助了解它是如何工作的。

TIGR 也提供了一个叫做 SpotFinder 的微阵列图像分析的 Windows 应用程序。SpotFinder 可以处理大多数微阵列扫描仪产生的 TIFF 格式文件。产生与 TIGR 的 ArrayViewer 和其他微阵列分析软件兼容的输出。

可视化高维数据

微阵列的结果可能难于可视化。微阵列实验通常至少有 4维（x-位置，y-位置，荧光强度和时间）。直接的阵列图像散点图没有太多的信息价值。需要帮助从高维数据集提取特征并以感性的图像格式显示这些特征的工具。

TIGR 提供了一个叫做 ArrayViewer 的 Java 应用程序。当前 ArrayViewer 的功能集中在检测不同的表达基因和用图形格式显示不同的表达结果。 ArrayViewer 的参数可以调节到与任何大小的微阵列数据相吻合，它也可以配置为允许直接访问基本的基因序列数据和注释。它还提供了几个标准化的选项。时间系列数据和其他更复杂

的数据集分析特征还没有实现，但是 ArrayViewer 的确可以满足大多数基本的阵列可视化需要。

一些普通的可视化和数据开采软件包，如我们将在第十四章中讨论的Xgobi软件包，也可以用于检查微阵列数据。

聚类表达简图

在写作本书时，最流行的微阵列数据分析策略是表达简图的聚类。表达简图可以以散点图的形式可视化。该散点图描述在实验过程中微阵列栅格中的一个点的表达变化。实验过程随着前后关系而改变，也就是随培养基中营养成分的浓度改变引起的任何情况而改变，在将细胞DNA与阵列进行杂交之前，细胞要在培养基中生长到一定的细胞周期相。

在这种前后关系中，被聚类的本质上是散点图的形状。不同的聚类方法，如等级聚类或者 SOM 法（self-organizing map ，自组织作图）可能在不同的条件下会更好，但这些方法的一般目的是相同的（注 5）如果在随环境改变的反应中两个基因以同样的方式改变表达水平，可以推测这些基因是相关的。它们可能是简单地共用一个启动子，或者更可能的是，它们被同一个复杂的调节途径所控制。表达简图的自动聚类寻找相似的表征（相似外形的表达简图），但是不必指出这些变化的原因。至少现在那是分析这些结果的科学家要做的工作。

程序 Cluster 和 Treeview，也来自斯坦福，它们是一个聚类表达简图的 Windows平台工具，实现了SOM 和等级聚类法等各种聚类算法。XCluster 实现了大多数 Cluster 的特征，可用于 Unix 平台。所有这些程序需要特殊的文件格式（详解见在线使用手册）。

关于表达分析商业软件的说明

现在可以得到带有可视化复杂的微阵列数据集工具的几种商业软件包。它们中的很多都专门用于特殊的硬件或者阵列配置。另外一些，如 SpotFire和 Silicon Genetics 的 GeneSpring则更通用一些。这些软件包常常需要昂贵的使用许可协议。然而，在微阵列技术发展的现阶段，它们相对简单的使用方法可能会使它们物有所值。

注 5:　在第十四章我们将较详细地讨论聚类方法。

蛋白质组学

蛋白质组学（proteomics）是指同时研究细胞整个蛋白质编制的技术。尽管蛋白质纯化和分离的方法正在不断提高，通过核磁共振和X射线晶体学测定蛋白质结构的完成时间也在变短，但是还没有快速结晶生物的整个蛋白质编制和决定每一个结构的单一方法。另一方面，生物化学定性技术正在变得更好更快。最需要信息学支持的生物化学的技术进步是固定梯度的2D-PAGE方法和随后通过质谱对分离蛋白质产物的定性。微阵列机器人已经开始用于产生蛋白质阵列，它可以用于蛋白质相互作用的分析、药物发现和其他应用。然而，蛋白质阵列还远不是常规的方法。

蛋白质组学实验方法

知道基因什么时候表达和表达到什么水平只是理解基因组如何决定表现型的第一步。虽然细胞中mRNA的水平和蛋白质浓度相关，但是蛋白质容易进行翻译后修饰，这不能用杂交实验检测。决定细胞中蛋白质浓度和活性的实验工具是这个方法的下一个关键步骤。

作为功能基因组工具，正在出现的另一个高通量技术是2D凝胶电泳。凝胶在分子生物学中用于混合组分的分离已经很长时间了。根据实验条件和使用凝胶的不同类型，不同的组分以不同的速率迁移通过凝胶介质（同样的原理使DNA测序成为可能）。

二维凝胶电泳可以用来分离含有上千种组分的蛋白质混合物，实验的第一维是沿着pH梯度分离溶液的组分（等电聚焦）。第二维是按分子量正交地分离组分。在这两个方向上的分离甚至可以解析复杂组分的混合物。图11-5显示了一个来自于拟南芥的二维聚丙烯酰胺凝胶电泳的参考图。二维聚丙烯酰胺凝胶电泳从混合样品中分离蛋白质使得每一个蛋白质可以被检测。图上的每一个点代表不同的蛋白质［图像版权归瑞士日内瓦瑞士生物信息学研究所（Swiss Institute of Bioinformatics, Geneva, Switzerland）所有］。

虽然二维凝胶电泳本身是一个有用而且令人感兴趣的技术，但是这项技术直到开发了标准化的固定梯度凝胶才真正流行起来。这些凝胶允许非常准确的蛋白质分离，产生标准化的高密度数据阵列。因此它们可以提交给自动化的图像分析仪并定量，用于准确的比较研究。将二维凝胶技术推到现代分子生物学方法前沿的其他进展是

使用质谱对凝胶上的每一点进行化学分析的能力。这允许可测量的生物化学现象（凝胶上特定点发现的蛋白质量）直接和该点发现的蛋白质序列联系起来。

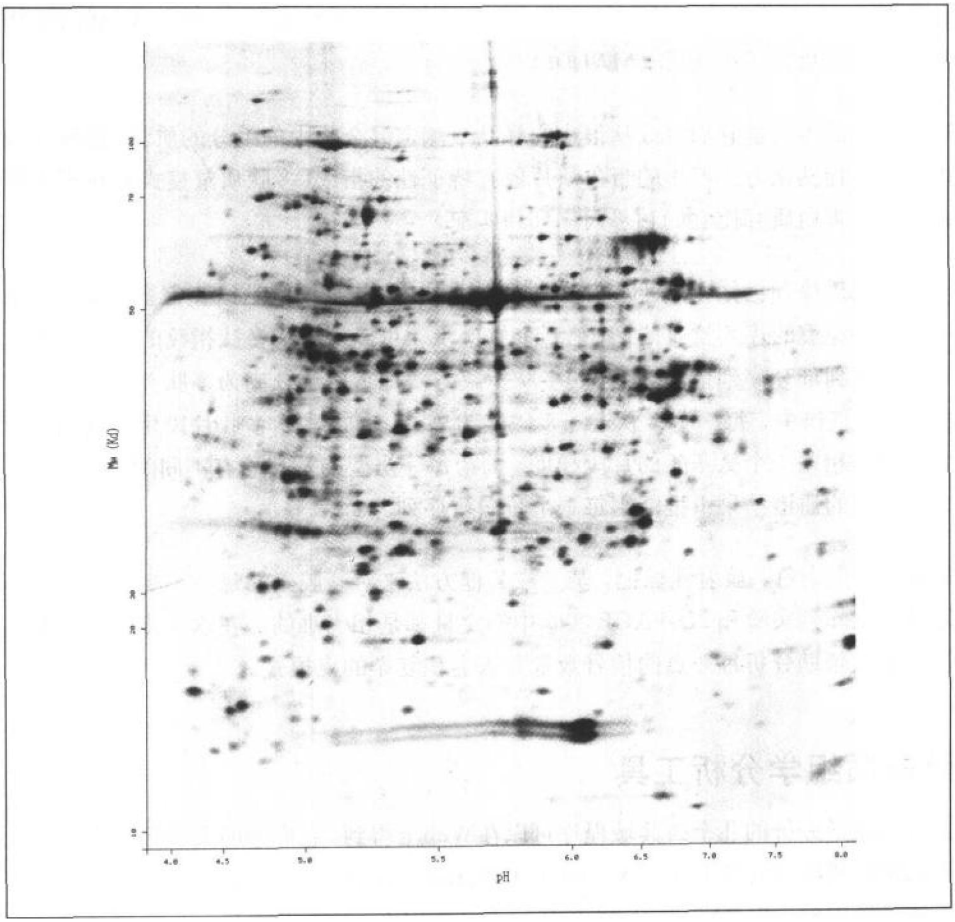


图11-5　来自拟南芥的二维聚丙烯酰胺凝胶电泳图谱

2D-PAGE 分析的信息学挑战

2D-PAGE 凝胶图像的分析途径基本上和微阵列的分析相似。第一步是图像分析，凝胶上点的位置被确定，不同点之间的边界被解析。凝胶上每一个蛋白质的分子量和等电离点（PI）可以根据位置来推测。

下一步，点被确定，序列信息用来建立特定点和它的基因序列间的联系。在微阵列实验中，随着引物或 cDNA片段在最初的芯片设计中被放置，这一步也会预先计划。在蛋白质组分析中，固定的蛋白质或者被原位测序，或者用机械方法从凝胶上切下蛋白质点，洗脱，使用质谱方法进行分析，如静电离子化质谱（ESI-MS）或者底物辅助激光解离离子化质谱（MALDI）。

质谱方法的本质是它们可以从很少的样品中测定混合物中的组分的质量。通过各种化学特异性消化方法产生的蛋白质片段有特征性的指纹（多肽质量模式）,可用来鉴定特殊的蛋白质并使它们和基因序列相匹配。

只要蛋白质序列已知，并可以在数据库中找到，多肽指纹就足以鉴定蛋白质。如果不能得到完整的序列信息的话，第二个质谱步骤可以从组成多肽指纹的每一个独立的多肽得到部分序列信息。最初的多肽指纹方法把蛋白质分离为多肽并通过质量定性。在质谱仪中，每一个多肽进一步破碎为离子化的片段。多肽片段化的目的是产生长度只相差一个氨基酸的片段梯度。因为每一类型的氨基酸有不同的分子量，可以从得到的质谱结果中推断出每一个蛋白质序列。

最后，使用染色、放射性标记、荧光或其他方法来对凝胶上的每一个蛋白质点进行定量。微阵列实验和 2D-PAGE 实验中的定量都是相当难的。在这一步中，计算机算法可以帮助分析每一点的信号数量并去卷积复杂的点模式。

蛋白质组学分析工具

蛋白质组学分析的几个公共域程序可以在 Web上得到。它们中的大多数可以通过优秀的蛋白质组学资源 ExPASy（Expert Protein Analysis System，专家蛋白质分析系统）访问到，ExPASy（*http://www.expasy.ch/tools/*）是由瑞士生物信息学研究所维护的优秀资源。ExPASy被链接到 SWISS-PROT（一个专家维护的蛋白质序列信息数据库）和 TrEMBL（计算机产生的 SWISS-PROT 的对应物）。它的工具的大多数是基于 Web 的，并与这些或其他一些蛋白质数据库绑定。瑞士生物信息学研究所也维护着 SWISS-2DPAGE,一个参考凝胶图的数据库，它是完全可搜索的并整合有其他蛋白质信息。SWISS-2DPAGE 与其他生物数据库一样增长迅速；然而，目前把2D-PAGE的结果放进数据库中对大众来说是不需要的，因此数据库是不全面的。

Melanie3软件包是由 ExPASy开发的基于 Windows 的 2D-PAGE 图像分析软件包，但是它已经商品化了。Melanie查看器是 ExPASy免费发布的，它允许用户不需要拥有 Melanie3 就可以查看同事生成的 Melanie3 数据集。

下面是其他一些 ExPASy 蛋白质组学工具:

AACompIdent
　　允许通过蛋白质的氨基酸组成鉴定蛋白质

AACompSim
　　比较一个蛋白质的氨基酸组成和 SWISS-PROT 中的其他蛋白质组成

MultiDent
　　一个使用 PI、分子量、质量指纹和其他数据帮助鉴定蛋白质的多功能工具

PeptIdent
　　比较实验测定的质量指纹和 SWISS-PROT中所有蛋白质的理论计算的质量指纹

FindMod
　　根据实验和计算的指纹质量的不同预测蛋白质的翻译后修饰

GlycoMod
　　按质量的不同预测寡糖修饰

PeptideMass
　　为SWISS-PROT 或者 TrEMBL 条目，或者为用户输入的蛋白质序列计算理论质量指纹。

这些工具是完全基于表单的，对初学者很适用。而且， ExPASy 提供了许多到外部开发工具和 Web服务器的链接。对于任何对蛋白质组学感兴趣的人来说，它都是一个优秀的起点资源。

PROWL数据库是一个相对较新的蛋白质组学网络资源。PROWL 工具可以用于使用多肽指纹或部分序列信息搜索蛋白质数据库。 PROWL 工具组还提供了一套质谱数据分析的工具。

推广阵列方法

将微阵列和 2D-PAGE 方法（它们分别提供有关基因转录和翻译的信息）与基因组序列信息整合是目前可以使用的建立整个基因组功能图景的最好方式。然而，这些方法仍然相当新。虽然技术在跳跃性地向前发展，但是它们的结果还不能完全标准化，兼容这些数据类型的软件工具和分析方法正在开发中。

阵列和 2D-PAGE实验有常见的要素，包括在混合物中分离和定量组分，并不断地固定特殊的组分到栅格中的位置的能力，在整个时间里测量每一点的信号变化的能力。无论是有关 DNA-DNA，DNA-mRNA ,还是甚至蛋白质 - 蛋白质间相互作用的实验，分析阵列格式的生物化学数据的方法可能都有某种程度的相似性。阵列策略最近也用于处理酵母中基因组范围的蛋白质 - 蛋白质间的相互作用的调查，这个策略的其他应用无疑也正在使用之中。阵列方法和其他平行方法有望可以继续生物学的革命。然而，生物团体仍然正在开发报告和获取阵列数据的标准，但在本书出版之前达到一致似乎是不可能的。

生化途径数据库

基因和蛋白质的表达如图 11-6所示（图像版权由 Kyoto Encyclopedia of Genes and Genomes 所有），仅是遗传密码转变成表现型中的两个步骤。一旦基因表达并翻译成蛋白质，它们的产物就参与到被称为途径（ pathway）的复杂的生物化学相互作用中。靠一个酶催化的化学反应从生物易于得到的物质中生产出需要的产物是非常不可能的。相反地，这往往需要一系列复杂的步骤。每一个途径可能为其他途径提供化学前体，这意味着每一个蛋白质不仅会和一个单一途径中的前后生化步骤有关，而且可能会和几个途径中的步骤有关。复杂的代谢途径分支比线性的基因和基因组序列更加难以表示和搜索。

复杂代谢途径的说明

几个基于 Web的服务可以提供对代谢途径信息的访问。这些资源都是用搜索工具链接的信息的初级数据库；在写作本书时，代谢模拟工具，像我们在下节讨论的那些，还没有把已知的代谢途径信息数据库完全整合成一个基干 Web 的中央资源。

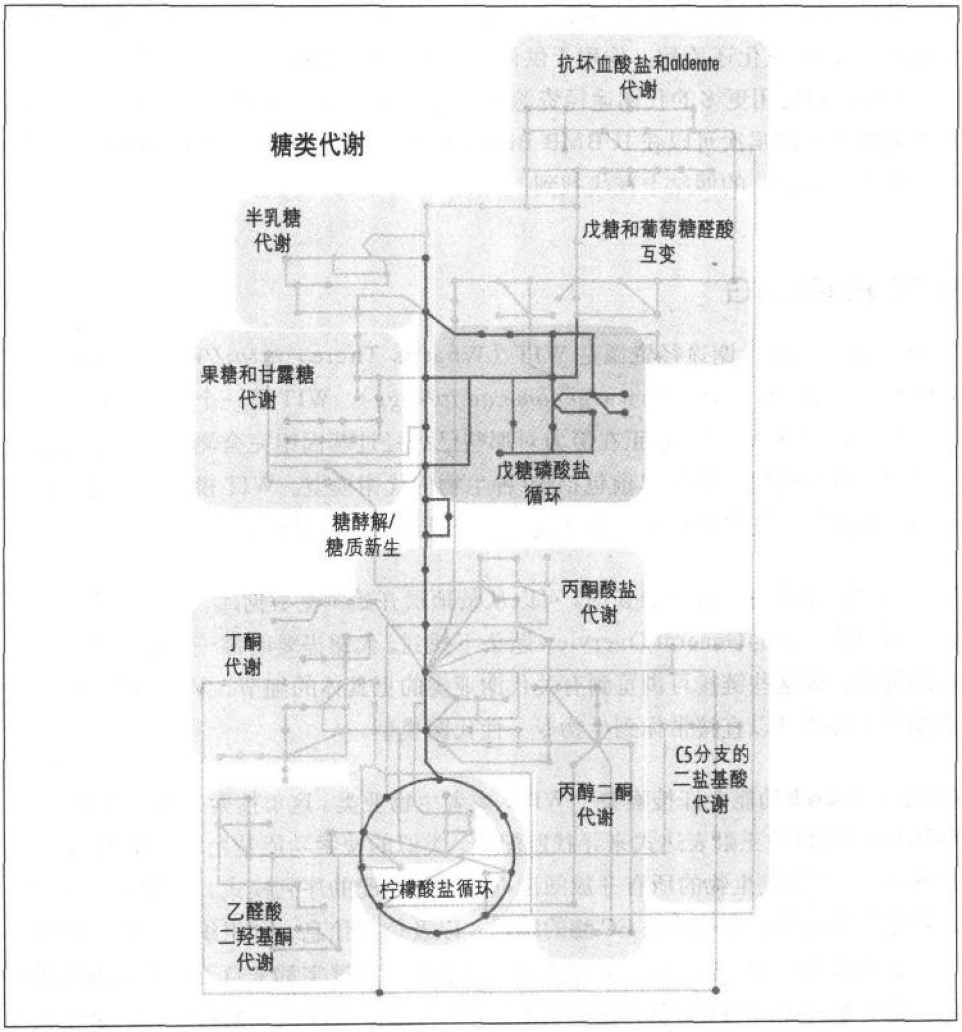


图 11-6　KEGG 描述的复杂的代谢途径

EC 命名法

酶（催化代谢反应的蛋白质）可以使用称作EC 编码的标准编码来描述。EC 命名法是一种分级命名方案，它把酶分成几种主要的类别。第一个分类数字是指酶的化学性质：氧化还原酶、裂解酶、脱氢酶、转移酶、异构酶和连接酶。第二个分类数字表示酶作用底物的分类。第三个分类数字可以省略，它表示反应中其他的化学参与

者。最后，第四个数字限制对特定酶的搜索。因此 EC 数字 1.1.1.1 指乙醇脱氢酶，它是一个（1）氧化还原酶，作用于供体（1）CH-OH集团，以（1）NADH 作为受体。如果你对使用更多的代谢途径资源感兴趣，熟悉 EC命名法是有帮助的。EC编码和功能定义的层次可以在 IUBMB Biochemical Nomenclature Committee （生物化学命名委员会）的网站上在线找到。

WIT 和 KEGG

Web上最著名的代谢途径资源是 WIT（What is There，*http://wit.mcs.anl.gov/WIT2/*）和 KEGG（*http://www.genome.ad.jp/kegg*）。WIT 是一个代谢途径重建资源；也就是，WIT 的管理者正在努力对那些已经经过基因组完全测序的生物的完整代谢途径进行重建。WIT 目前包括 39 种生物的代谢模型。WIT 模型远不仅包括代谢和生物能学，还包括转录、翻译途径、跨膜运输和信号转导。

WIT可以以多种方式搜索和查询。可以从最顶层开始浏览数据库，WIT 内容的功能概况可以在主页的 General Overview 题头下找到。代谢提要的每一个题头都是可点击的链接，到这些链接可浏览到有关代谢亚类的更具体的细节。 View Models（查看模型）菜单可以直接带你到生物专一性代谢模型。

General Search 功能允许搜索整个WIT，或者生物亚类。这类搜索是基于关键字的，利用 Unix 风格的正则表达式来寻找匹配项。类似的搜索功能是允许使用 BLAST 或者 FASTA 在选定生物的所有开放阅读框中搜索匹配的序列模式。代谢途径查询要求指定代谢物的名称和特殊 EC 酶的编码，以及位置信息，如组织专一性、细胞器专一性或者器官专一性。除了正则表达式搜索外，关键字都来自于标准化的代谢词汇。提交查询时， WIT需要有这些词汇的一些知识。 WIT最初的设计目的是作为工具帮助开发者产生代谢重建。使用词汇的文档资料对初学者可能已经足够了。虽然在最详细的水平上可以显示代谢途径示意图，但是 WIT相对还是以文本为主。

另一个基于 Web 的代谢重建资源是 KEGG，它使用图例说明的方式提供了代谢概况，而不仅仅是文本，对于视觉导向用户来说，它更容易。 KEGG还提供了一个 EC 编号的列表，它们对应的酶被分段分解，许多有用的链接可以连接到详细描述酶和配体命名的站点。LIGAND 数据库和 KEGG结合在一起，是鉴别生化途径中的相关小分子的有用资源。与 WIT 一样，KEGG 可以通过序列同源性、关键字、化学条目

等进行搜索；也可以输入两个小分子的 LIGAND ID 号，找到连接它们的所有可能的代谢途径。

PathDB

PathDB是另一种类型的数据库。虽然它大体上包含和KEGG与 WIT相同的信息(相同的化合物和代谢酶，以及相同的连接这些实体的步骤信息），但是它使用比其他数据库更灵活的方式处理信息。没有限制对任意的代谢途径的搜索，也没有描述预想图像的途径，PathDB 允许寻找到任何一组相连的反应，它们或者连接着 AB 两点，或者是连接 AB两个化合物。

除了通常的检索工具外，PathDB还包括途径可视化界面，它允许观察选择的任何途径，并显示途径的不同表示法。PathDB开发者也想把用户模拟整合进用户界面，虽然在写作本书这些特征还没法得到。

PathDB 浏览器是一个独立于平台的工具，可以用于安装了 Java Runtime Environment（JRE）1.1.4 或更新版本的任何机器上。Sun和 IBM 都为 Linux 系统提供了 JRE。一旦 JRE 安装后，就可以运行 PathDB 安装程序，确信安装程序使用了正确的 JRE 版本（为了这样做，可能需要把 JRE二元目录加到你的路径中)。让安装程序在主目录中建立一个到 PathDB 的链接，输入 PathDB，即可运行该程序。你可能被提示指定正确的 Java虚拟机否则退出；使用与你安装软件时相同的 Java路径。

为了用例子显示 PathDB是如何工作的，提交一个简单的查询，它可以使你确信在第一时间得到结果（例如“所有 EC编码类似 1.1.1.1 的催化剂”,它会带来一个乙醇脱氢酶的清单）。你也可以按照 PathDB网站的指导手册进行。

动力学和生理学建模

一个最近出现在文献中的新的“omics”时髦词语是“代谢组学(metabolomics)”。研究者已经开始认识到需要不遗余力地追踪生物系统中的小分子（从电解质到代谢中间物到酶辅助因子的一切物质）的有效性和浓度。小分子在生物化学中是非常重要的，在从信号转导到生物能，到调节的所有事情中都发挥着作用。随着生物学家

努力收集细胞功能和生理的大的图片，细胞中和蛋白质相互作用的小分子群体将继续是关键的研究课题。

生物化学动力学和生理学的数学建模是一个复杂的课题，已远远超过了本书讨论的范围。数学模型通常是系统专一性的，开发它们需要对生物系统和不同方程的功能有详细的了解。然而，几个小组已经开始开发用于生物化学和生理学模型的前后关系非依赖性的软件。它们当中最著名的一些是 Gepasi（一个生物化学建模系统）；XPP （一个更常用的动力学模拟软件包）和 Virtual Cell portal（虚拟细胞入口）。

生物化学和生理学建模背后的基本原理是生物化学系统的变化可以根据化学浓度和相关的速率等式进行建模。系统中的每一个生化试剂“池”都有一个相关的生成和降解速率。模型能够预测在各种起始条件下，在一段时间内系统将会如何行动。代谢模型可能包括几十种试剂，每一种都由多个反应生成和消耗。精确模拟复杂生物化学途径行为的模型不是很容易开发的，但是，一旦建立，它们就可以预测对系统的微扰效果并且可以帮助研究者发展新的假说。

如果有生物科学的背景，你可能熟悉描述酶动力学的 Michaelis-Menten模型。生物化学建模将这种相对简单的单酶催化反应的数学模型推广到在一个复杂途径中相互之间可能有反馈作用的多个反应中。生理模型也包括多个有边界的套间，只有一部分成分可以通过它扩散或转运。然而，无论模型如何复杂，基本原理都是相似的。

使用 Gepasi 建立动力学模型

Gepasi（*http://www.gepasi.org/*）是一个用户友好的 Windows/NT下的生物化学动力学模拟器，它可以模拟多达 45个代谢物和45 个速率等式的系统。Gepasi的界面包括创建新的代谢模型的交互式工具：输入化学反应、添加可能是反应促进子或者抑制子的代谢物、定义反应动力学、设置代谢物浓度和模型开发中的其他的关键步骤。可以将 Gepasi预设的反应类型应用到你的模型中或者定义你自己的反应类型。Gepasi自动检查模拟中需要遵守的质量守恒关系。 Gepasi有多个选项用于在各种时间过程中的模拟运行和测试用户定义范围内的可变值改变的结果。 Gepasi也可以优化用于代谢工程的代谢模型和使实验数据符合代谢模型。

在写作本书时，用于 Windows/NT 以外平台的 Gepasi 版本正在开发之中。

XPP

XPP（*http://www.math.pitt.edu/~bard/xpp/xpp.html*）是一个用于 Windows/NT 和 Linux 系统的动力学系统模拟工具。虽然它缺少一些 Gepasi 对用户友好的特征，但是它已经被有效地用于建立从生化反应、细胞周期到生理周期节律的生化过程的模型。XPP 可在任何 Linux 系统上使用简单的 *make*命令很容易地进行编译；仅仅下载压缩包，把它移到自己的目录中，解压缩，然后编译程序即可。文档资料和各种模拟的实例文件都包含在 XPP 的发布包中。

使用虚拟细胞入口

NRCAM（National Resource for Cell Analysis and Modeling，细胞分析和建模的美国国家资源 *http://www.nrcam.uchc.edu* ）的虚拟细胞入口（ Virtual Cell Portal）是第一个细胞过程建模的基于Web的资源。它允许使用任意数量的套间和复杂的生理学来建立细胞模型。虚拟细胞入口网站上的指导手册可以通过开发细胞生理学模型，选择细胞几何形状和建立并运行模拟的过程让第一次使用的用户熟悉应用方法。细胞生理学模型不仅包括细胞的分区方案，它可以使用简单的绘图工具生成，而且也包括为细胞模型添加特殊的离子种类和膜转运体。

虚拟细胞入口是一个 Java applet ，它完全支持苹果机和 Windows用户。为了在Linux 工作站上使用细胞虚拟入口，需要下载 Netscape的 Java插件（可从 *http://www.blackdown.org*得到），并安装到 *~/.netscape* 目录下。插件一旦安装，就可以按照主页上链接的“ Macintosh Users Run the Virtual Cell （苹果机用户运行虚拟细胞）”的提示进行，即使是在 Linux工作站上运行 VCell Applet，也能尝试入口的大多数性质。在写作本书时， Unix 用户还不能得到虚拟细胞入口的明确支持，尽管使用 Blackdown的 Java applet似乎可以得到正确的功能，但是对于严谨的 VCell工具用户来说，比较一些来自 Linux 工作站和其他机器的测试实例的结果是明智的。

小结

表 11-1是基因组学和蛋白质组学工具及技术的快速参考表。

基因组学和蛋白质组学工具及技术

目的	原因	可用工具
在线基因组资源	寻找基因组中特定基因的位置和功能信息	NCBI 工具、TIGR 工具、EnsEMBL和基因组专一性数据库
碱基命名	将测序实验的荧光强度转变成四字母的序列编码	Phred
基因组作图和组装	将短片段的原始 DNA 序列数据集织成连贯的整体	Phrap、Staden 软件包
基因组注释	把基因组的功能信息和专一的序列位置联系在一起	MAGPIE
基因组比较	鉴定一个生物和另一个生物的基因组间的不同的基因组结构成分	PipMaker、MUMmer
微阵列图像分析	鉴定和定量原始微阵列数据的点	CrazyQuant、SpotFinder、ArrayViewer
微阵列数据的聚类分析	鉴定可能一起表达的基因	Cluster、TreeView
2D-PAGE 分析	分析、可视化和定量 2D-PAGE 图像	Melanie3, Melanie Viewer
蛋白质组学分析	分析质谱结果和鉴定蛋白质	ExPASy 工具、ProteinProspector、PROWL
代谢途径工具	搜索代谢途径和发现功能关系；重建代谢途径	PATH-DB、WIT、KEGG
代谢和细胞模拟	根据已知的性质和推理建立代谢过程及细胞过程的模型	Gepasi、XPP、Virtual Cell

第四部分

数据库和可视化

第十二章，用 Perl 进行数据自动化分析
第十三章，构建生物学数据库
第十四章，可视化和数据采集

第十二章

用Perl进行数据自动化分析

从前面一些章节中我们已经了解到有很多软件可应用于生物信息学。尽管可能已有人编制出了我们需要的软件，但我们还是时常会碰到最好是自己解决问题的时候。在生物信息学中，这常常意味着自己编写从海量数据中提取出恰恰是需要的信息的程序。Perl（Practical Extraction and Reporting Language 的缩写，实用摘要和报告语言）是适合完成这种任务的理想语言。

为什么选择Perl?

现在已经有很多种编程语言，在对生物信息学软件的调查中，我们发现很多是用Java、C 和 FORTRAN 等编写的。既然这样，又为什么要使用 Perl 呢？答案是：为了提高效率（注 1）。生物学数据保存在大量数据库和文本文件中，手工排列和分析这些数据虽然可行但要耗费大量的时间，因此就有聪明的科学家编制了软件工具进行自动化操作。Perl具有检测数据模式的强大功能，尤其是字符串文本数据，因而是最明智的选择。另外一个理所当然的选择可能是Python。Python 与 Perl相比不太为人所知，它是由 Guido van Rossum 1988 年引进的完全面向对象的脚本语言。Python 有一些出色的共享代码，包括一个成熟的数字化方法库、快速而方便创建图

注 1：程序员们所说的效率，是指用 Perl语言处理数据比用 C 或Java 语言可节省大量的编程时间。

形用户界面的工具以及一个结构生物学功能库。然而，到目前为止，现行 Perl 代码是适用于生物信息学的宝库，代码和 Unix 系统的完美整合，跨平台的可移植性，以及狂热的用户群使 Perl 成为最方便的用于生物信息学的脚本语言。

Perl具有灵活的语法，如果熟悉其他语言如 C或 BASIC的编程，就很容易用类似 C 或 BASIC 的语言编写 Perl代码。Perl 还可以解决编程中的许多棘手问题，比如内存的分配等，这样我们就可以集中精力解决手边的问题。编程时，用 C 或 Java 语言需要很多行代码才能解决的问题，用 Perl 也许只需要很少的几行就能解决，这种例子比比皆是。

已有很多关于 Perl学习和应用的书籍，因此仅靠这一章，是不能全面介绍这种语言的。Perl博大精深，想不通过扎扎实实的系统学习就掌握计算机编程艺术是不现实的。本章的目的不是介绍如何用 Perl 编程，而是展示 Perl怎样解决生物信息学中的特定问题。我们将介绍生物信息学研究中最常遇到、最有用的例子，如读取数据文件、搜索字符串、完成后台计算、向用户报告结果等。我们将解释样例程序如何工作，剩下的由你来完成。对生物信息学家来说，能使用程序语言，特别是能使用 Perl、Python、Java 等语言来编程是需要掌握的一种非常重要的技巧。我们强烈建议你修一门编程课，或从我们列出的推荐读物中找一本书从头学起，只有这样你才不会感到遗憾。

从哪里得到 Perl?

Perl 可适用于多种操作系统，包括 Windows、MacOS、Linux 和 Unix 其他改进版。Perl 是用开源软件，这意味着 Perl 完全是免费的。从下面的网址可以得到 Perl，*http://www.perl.com/pub/language/info/software.html*，可按照说明下载并安装在你的系统里。

Perl 基础

一旦安装了 Perl，就可以准备开始编写你的第一个程序。编写和运行 Perl程序可以分成几个步骤：编程（或脚本）并保存在一个文件里、运行程序、读取输出结果。

Hello World

Perl程序是包含用Perl语言编写的指令的文本文件。“Hello, World!”是用Perl（以及其他很多语言）编写的第一个程序：

```
#!/usr/bin/perl -w
# Say hello
print "Hello, World!\n";
```

“Hello World!”这个程序虽短，却是一个完整的程序。第一行称为shebang行，表明这是一个Perl程序。所有在Unix中运行的Perl程序都以shebang行开始（注2），这是对Unix Shell 的一种特殊的注释，告诉它到何处去找Perl，同时指示其寻找可选参数。在我们的这个“Hello World!”程序中，在该行的结尾处包括可选参数 *-w*，如果你的操作可能危及程序的安全，这个参数就会告诉Perl发出特别的警告消息。在 *-w* 下编程是一个良好的习惯。

第二行以#开始，#表示随后的一行文本是注释，不属于可执行代码。注释用来告诉人们程序的每一部分要做什么。应养成在代码中加入注释的习惯，这样，你和其他人就可以很好地添加、调试代码，更重要的是，这样可以使你记住本来希望它做什么。

第三行称为print（打印）函数，该函数有一个由文本字符串组成的参数。在文本字符串的末端有一个 \n，告诉计算机执行 *print* 语句后换行。和Perl的大多数语句一样，*print*语句应以分号结束。

如果想亲自尝试一下这个小程序，可以打开一个文本编辑器，如 *vi*、Emacs或 *pico*，输入这几行。完成程序输入后，将文件命名为 *hello.pl*，并保存在你选定的目录中。在学习的时候不妨在你的家目录下创建一个名为Perl的目录（使用在第四章中介绍的 *mkdir*命令）。这样你就总能知道从何处能找到Perl程序了。

现在使用下面的命令将其变为可执行文件：

注2：严格地说，在Windows 或Macintosh操作系统中shebang行不是必需的；两者都没有*usr/bin/perl*目录；不过因为这里是显示Perl中可选参数的最佳位置，好的程序应包括这一行。在其他平台上，可以像在 *perl hello.pl* 中一样，通过显式调用Perl解释器来运行程序。

```
% chmod +x hello.pl
```

（如果需要重温 *chmod*，这正是一个温习第四章中改变文件权限部分的好机会。）要运行这个程序，输入：

```
% hello.pl
```

因为在程序中存在 shebang行，这个命令会调用 Perl 解释器，阅读文件的剩余部分，然后将 Perl 源代码转换成计算机可以执行的机器代码。这样你将看到计算机屏幕上出现“Hello，World!”，然后光标移至下一行。现在你已经编写并运行了你的第一个 Perl 程序！

一个生物信息学的例子

Perl的一个优势（也是生物信息学家喜爱它的原因）是仅用几行代码就可以使一些冗长乏味的工作自动化，例如从一大堆序列数据中搜索核酸序列。为此，需要类似这样的稍微复杂些的 Perl 程序：

```
#!/usr/bin/perl -w
# 在序列数据中寻找字符串
my $target = "ACCCTG";
my $search_string =
    'CCACACCACACCCACACACCCACACACCACACCACACACCACACCACACCCACACACACA'.
    'CATCCTAACACTACCCTAACACAGCCCTAATCTAACCCTGGCCAACCTGTCTCTCAACTT'.
    'ACCCTCCATTACCCTGCCTCCACTCGTTACCCTGTCCCATTCAACCATACCACTCCGAAC';

my @matches;

# 试图寻找$search_string 1~6 个字母的匹配，然后看 2~7个字母的
#  依此类推。记录每个匹配的起始偏移

foreach my $i (0..length $search_string){
   if( $target eq substr( $search_string, $i, length $target)){
      push @matches, $i;
   }
}

# 将 @matches放入逗号分开的列表中输出
print "My matches occurred at the following offsets: @matches.\n";
print "done\n";
```

这个程序也较短，而且比较简单，但功能仍十分强大。它从数据序列中搜索靶序列“ACCCTG”，并给出每个区配的起始位置。这个程序演示了变量和循环，这是我们需要理解的程序的两个主要构造。

变量

变量（variable）是与数据值如字符串或数字有关的名称。通常说变量储存或包含值。变量允许在程序中储存或操作数据；它们之所以称为变量，是因为它们所代表的值在程序运行的全过程中是可变的。

我们的序列匹配程序声明了四个变量：*$target*、*$search_string*、*@matches*和*$i*。$和@表明了每个变量的类型。Perl语言中有三种变量：标量（scalar）、数组（array）、散列（hash）。

和其他编程语言不同，Perl不要求正式的变量声明；无论是否显式声明这些变量，它们在第一次使用时就存在了。如果愿意，可以在变量名称前使用*my*或*our*来声明变量。声明变量时要为其命名。变量名称必须遵循两条主要规则：必须以字母或下划线开始（$和@字符不是名称的组成部分）；必须由字母、数字和下划线组成。最好的变量名称应该能够清楚、简明而准确地描述变量在程序中所起的作用。例如，如果一个变量被命名为*$target*或*$sequence*，要比称为*$icxl*更容易猜出它的作用。

标量

标量变量（scalar variable）包含一个简单的数据，可以是数字或字符串。$表示变量是标量。我们的程序中声明的前两个变量就是标量变量：

```
my $target = "ACCCTG";
my $search_string =
    "CCACACCACACCCACACACCCACACACCACACCACACACCACACCACACCCACACACACA".
    "CATCCTAACACTACCCTAACACAGCCCTAATCTAACCCTGGCCAACCTGTCTCTCAACTT".
    "ACCCTCCATTACCCTGCCTCCACTCGTTACCCTGTCCCATTCAACCATACCACTCCGAAC";
```

这样，“ACCCTG”就是我们要搜索的靶序列，“CCACACCACACCCACAC…”则为我们希望在其中找到靶序列的序列数据。

在标量变量中，数字可以是整数（0、1、2、3等）或是实数（含有小数部分的数字，如5.6）。字符串是用引号包括的一段文本。例如：

```
"I am a string."
'I, too, am a string
```

Perl的一个特征就是具有很多用来操作字符串的内置工具，这在操作生物信息学中常见的平面文本文件时很有用。我们将在第十三章中详细讨论平面文本文件及结构'ㄴ程度更高的关联数据库。

数组

数组（array）是一个有序的数据列表。在我们的序列匹配程序中，*@matches*是一个储存所有匹配序列起始位置的数组变量。数组中储存的每一个元素都可以通过它在列表中的位置访问，其位置是用数字代表的。在 Perl 中，数组变量用@作为前缀。例如，下面的语句声明了一个数字数组：

```
@a = (1, "4", 9);
```

以下语句声明一个字符串数组：

```
@names = ("T. Herman", "N. Aeschylus", "H. Ulysses", "Standish"
```

以下语句声明一个包含数字和字符串的数组

```
@mix = ("Caesar Augustus", "Tiberius", 18, "Caligula", "Claudius"
```

请注意声明的语法：数组中的每一个元素之间用逗号隔开，每个字符串加引号，而且逗号位于引号之外（不同于美国英语）。

由于数组是一组有序的信息，可以根据相应的数字检索数组中的每一个元素。数组中的单个元素写成：*$this_array[i]*,其中 *i* 是数组元素的索引。*i*既可是一个裸数（如 *21*），也可是包含裸数的数字标量变量（如 *$n*）。以下是一个用 *print* 运算符在屏幕上显示 *@a* 中第二个数字和 *@names* 中第三个名字的 Perl 语句：

```
print "second number: $a[1]\n third name: $names[2]\n"        ;
```

你可能会产生疑问，为什么这里的元素数字比你认为的数字要小 1。原因就是 Perl 数组中的位置是从 0 开始编号的。也就是说，数组中的一个元素编号为 0、第二个元素编号为 1，依此类推。这就是在前面的例子中 *@a* 中的第二个元素被编址为＄*a[1]* 的原因。这是一个应该牢记的细节，否则很容易在编址数组时犯由于遗漏了那个关键的 0 元素而导致的错误。

Perl的一个特征就是具有很多用来操作字符串的内置工具，这在操作生物信息学中常见的平面文本文件时很有用。我们将在第十三章中详细讨论平面文本文件及结构化程度更高的关联数据库。

数组

数组（array）是一个有序的数据列表。在我们的序列匹配程序中，*@matches*是一个储存所有匹配序列起始位置的数组变量。数组中储存的每一个元素都可以通过它在列表中的位置访问，其位置是用数字代表的。在Perl中，数组变量用@作为前缀。例如，下面的语句声明了一个数字数组：

```
@a = (1, "4", 9);
```

以下语句声明一个字符串数组：

```
@names = ("T. Herman", "N. Aeschylus", "H. Ulysses", "Standish"
```

以下语句声明一个包含数字和字符串的数组

```
@mix = ("Caesar Augustus", "Tiberius", 18, "Caligula", "Claudius"
```

请注意声明的语法：数组中的每一个元素之间用逗号隔开，每个字符串加引号，而且逗号位于引号之外（不同于美国英语）。

由于数组是一组有序的信息，可以根据相应的数字检索数组中的每一个元素。数组中的单个元素写成：*$this_array[i]*,其中*i*是数组元素的索引。*i*既可是一个裸数（如*21*），也可是包含裸数的数字标量变量（如*$n*）。以下是一个用*print*运算符在屏幕上显示*@a*中第二个数字和*@names*中第三个名字的Perl语句：

```
print "second number: $a[1]\n third name: $names[2]\n"        ;
```

你可能会产生疑问，为什么这里的元素数字比你认为的数字要小1。原因就是Perl数组中的位置是从0开始编号的。也就是说，数组中的一个元素编号为0、第二个元素编号为1，依此类推。这就是在前面的例子中*@a*中的第二个元素被编址为$*a[1]*的原因。这是一个应该牢记的细节，否则很容易在编址数组时犯由于遗漏了那个关键的0元素而导致的错误。

```
    g464769  => "SODC_CAEEL",
    g1711426 => "SODC_ECOLI" );

# 从%sods 检索值
$genbank_id = "g134611";
$swissprot_id = $sods{$genbank_id};
print "$genbank_id is the same as $swissprot_id\n";
```

如果将前面的脚本保存成一个文件，并将其转变为可执行文件，运行，将看到：

```
g134611 is the same as SODC_HUMAN
```

在脚本的第一部分，声明一个将GenBank ID和SWISS-PROT ID相关联的散列。在第二部分，访问存储在这个散列中的信息。第一步是用一个GenBank ID给变量*$genbank_id*赋值。然后检索 *%sods*与*$ genbank_id*中的字符串相关联的SWISS-PROT ID。最后，打印这两个标量变量的值。这个例子不一定符合实际情况，但它能给你一个散列在生物信息学程序中起着重要作用的概念。

循环

我们已经讨论了 Perl中的标量、数组和散列变量，现在让我们回到序列匹配程序，讨论在其中使用的其他主要编程构造。循环（loop）是重复执行特定的一组命令直至达到某一条件的编程方法。我们的程序使用*foreach*循环对搜索字符串进行迭代。

```
foreach my $i (0..length $search_string){
   if( $target eq substr( $search_string, $i, length $target))
      push @matches, $i;
   }
}
```

第一次循环，Perl从0开始并查看搜索字符串的第一个六字母组合，将其与靶字符串进行比较，如果匹配，记录在 *@matches* 中。第二个循环查看第2~7个字母，第三个循环查看第3~8个字母，依此类推。当到达搜索字符串的结尾时， Perl停止执行这个循环。在这 !!! 循环完成，程序进入下一部分，打印结果。如果并不理解循环中所有的代码，先不必担心；现在重要的是应对代码在执行什么有大体的了解。

子例程

尽管我们在实例程序中并没有用到子例程，但它在程序中的使用却是值得讨论的。

所有的现代编程语言都提供了将一组语句捆绑成子例程使之能被简洁、重复调用的方法。在 Perl 中，可以用 *sub* 声明创建一个子例程：

```
sub greet {
    my ($name) = shift;
    print "Hello, $name!\n";
}
```

一旦声明了这个 *greet* 子例程，就可以像下面这样调用：

```
greet("world");     # 输出 "Hello, world!"
greet("Per");       # 输出 "Hello, Per!"
```

这里，"world"和"Per"是参数（argument）——传递给子例程的值，这里它们存储在 *$name* 中。我们的 *greet* 子例程仅打印单行，然后返回。通常，子例程执行稍微复杂些的任务，并可能返回值：

```
$length = calculate_length($sequence);
```

提供单参数 *$sequence* 时，将 *$length* 设置为 *calculate_length()* 子例程返回的值。当一个子例程用于得到返回值时，常常被称作函数（function）。

模式匹配和正则表达式

Perl 的一个主要特征是它的模式匹配，尤其是它对正则表达式的使用。正则表达式（在 Perl 中称作 *regex*）是一种与数据字符串匹配的模式。我们在第五章中讨论 Unix 命令 *grep* 时见到过正则表达式。大家可能记得，*grep* 是用来在文件中搜索某一模式的命令。告诉 *grep* 搜索一个模式时，即按照正则表达式描述所要寻找的对象。正如你所知道的，生物信息学中很多是关于在数据中搜索模式的。

让我们来看一个 Perl 的例子。假设有一个字符串，如一个 DNA 序列，我们想确定其中没有非法字符，可以像下面这样用正则表达式检查非法字符：

```
#!/usr/bin/perl
# 检测非 DNA 字符
my $string =
"CCACACCACACCCACACaCCCaCaCATCACACCACACACCACACTACACCCA*CACAC
ACA";
if( $string =~ m/([^ATCG])/i) {
  print "Warning! Found: $1 in the string";
```

这个程序包含正则表达式*[^ATCG]*。用文字来描述这个正则表达式，就是‘在*$string*中寻找与 A、T、C 或 G 不匹配的字符”（语句结尾处的 *i* 是告诉 Perl 不区分字母大小写。Perl 默认的是区别 A 和 a）。如果 Perl 遇到的与已声明的模式不同，程序打印 offending 字符。程序输出：

```
Warning! Found * in the string
```

如果改为搜索一个特定的字母组合，如“CAT”，可以把正则表达式改为读 *CAT*:

```
#!/usr/bin/perl
# 检测CAT
my $string =
    "CCACACCACACCCACACaCCCaCaCATCACACCACACACCACACTACACCCA*CACACACA";
if( $string =~ m/CAT/i ){
  print "Meow.";
}
```

修改后程序输出：

```
Meow.
```

使用 Perl 解析 BLAST 输出

现在大家对 Perl 是怎样编写的认识足以理解这些简单的程序。我们来把它应用到生物信息学的最常见问题之一中去：解析 BLAST 输出。我们已经知道，BLAST 搜索的结果常常是一个很多兆字节的充满原始数据的文件。几次搜索的结果很快就会变得令人无所适从。但是用 Perl 编写一个相当简单的程序，就可以使得从数据中寻找一个单数据字符串或多数据字符串的过程自动化。

考虑下面的数据块：

```
gb|AC005288.1|AC005288  Homo sapiens chromosome 17, clone hC...   268  2e-68
gb|AC008812.7|AC008812  Homo sapiens chromosome 19 clone CTD...   264  3e-67
gb|AC009123.6|AC009123  Homo sapiens chromosome 16 clone RP1...   262  1e-66
emb|AL137073.13|AL137073  Human DNA sequence from clone RP11...   260  5e-66
gb|AC020904.6|AC020904  Homo sapiens chromosome 19 clone CTB...   248  2e-62
>gb|AC007421.12|AC007421 Homo sapiens chromosome 17, clone hRPC.1030_O_14,
complete sequence
Query: 3407 accgtcataaagtcaaacaattgtaacttgaaccatcttttaactcaggtactgtgtata 3466
```

```
                 ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct: 1366 accgtcataaagtcaaacaattgtaacttgaaccatcttttaactcaggtactgtgtata 1425
Query: 3467 tacttacttctcccccctcctctgttgctgcagatccgtgggcgtgagcgcttcgagatgt 3526
                 ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct: 1426 tacttacttctcccccctcctctgttgctgcagatccgtgggcgtgagcgcttcgagatgt 1485
Query: 3527 tccgagagctgaatgaggccttggaactcaaggatgcccaggctgggaaggagccagggg 3586
                 ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct: 1486 tccgagagctgaatgaggccttggaactcaaggatgcccaggctgggaaggagccagggg 1545
Query: 3587 ggagcagggctcactccaggtgagtgacctcagccccttcctggccctactcccctgcct 3646
                 ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct: 1546 ggagcagggctcactccaggtgagtgacctcagccccttcctggccctactcccctgcct 1605
Query: 3647 tcctaggttggaaagccataggattccattctcatcctgccttcatggtcaaaggcagct 3706
. . .
```

这仅仅是我们可能在 BLAST 搜索报告中发现的数据的一小部分。（这是来自一份BLAST报告的真实数据。*blast.dat* 的整个文件非常庞大，无法在这里再现。）这个例子的前六行包含 BLAST 搜索的信息，还有其他与搜索无关的“噪音”。下面 13行以及在实际报告中它的下一行包含要分析的数据。要求 Perl 程序检查 BLAST 报告中的“ Query ”行和“ Sbjct ”行，并得到下列子字符串出现的次数：

- 'gtccca'
- 'gcaatg'
- 'cagct'
- 'tcggga'

丢失数据（在核酸序列中用破折号表示）

同时，让程序忽略搜索信息和其他“噪音”等无关资料，然后程序应产生一个描述结果的报告文件 *report.txt*。

在这个程序中需要创建两个很长的标量变量来代表每一个要搜索的序列。我们把它们称为 *$query_src* 和 *$sbjct_src*。在任何 BLAST输出数据中都会发现，有时“Sbjct”行和“ Query ”行不连贯，即数据中有间隔。从编程的角度来看，存在间隔并不要紧；我们只是想把核苷酸连续地读入标量。这里是 BLAST 数据的示例：

```
Query: 1165   gagcccaggagttcaagaccagcctgggtaacatgatgaaacctcgtctctac 1217
              |||| ||||||||   ||||||||||||| |||| | |||||||  ||||||||
Sbjct: 11895  gagctcaggagtttgagaccagcctggggaacacggtgaaaccctgtctctac 11843
Query: 1170   caggagttcaagaccagcctg 1190
              |||||||||||||||||||||
Sbjct: 69962  caggagttcaagaccagcctg 69942
```

```
Query: 1106  tggtggctcacacctgcaatcccagcact 1134
             ||||||||||| |||| ||||||||||||
Sbjct: 77363 tggtggctcacgcctgtaatcccagcact 77335
```

尽管行号不连续，“Query ”的序列仍是从“ gagccca ”开始，到“agcact ”结束，长 103（53+21+29）个字符。不久我们就会看到，程序被设计成忽略间隔（和行号），数据被正确地输入。常用 BLAST 的人可能还会注意到，在完整的 BLAST 报告中，每个序列按照 E- 值分组。我们在程序中忽略了这个通常较为重要的事实。

搜索这 5 个子字符串的 Perl 程序可以分解为三部分：

输入数据，准备分析

搜索数据，查找模式

编译结果，存入 report.txt

让我们来一步步地研究这个程序。这里是前几行：

```
#!/usr/bin/perl
# 搜索一个大数据文件，寻找特殊序列

use strict;

my $REPORT_FILE = "report.txt";
my $blast_file  = $ARGV[0] || 'blast.dat';

unless ( -e $blast_file ) {
  die "$0: ERROR: missing file: $blast_file";
}
```

这部分程序可确保数据排列整齐。由于要向变量中读入大量数据，用 *use strict*; 行告诉 Perl 使其规则更严格。这就迫使你更明确你想让 Perl 做什么。在开发大型程序或修改和重复使用程序时，*use strict* 尤其有效。继续声明一些变量，在最后几行告诉 Perl 确认数据存在于输入文件 *blast.dat* 中。

在下一部分，程序把序列读入变量：

```
# 首先将所有 Query序列放入一个标量
# Sbjct 序列也一样
my ( $query_src, $sbjct_src);

# 打开 blast数据文件，如果找不到，结束程序 ( die)
```

```
open (IN, $blast_file) or die "$0: ERROR: $blast_file:

# 逐行遍历 blast 文件，将所有 Query
# 与Sbjct 序列合并
while (my $line = <IN>) {
  chomp $line;
  print "Processing line $.\n"   ;
```

在这一部分，把全部“ Query ”序列读入一个标量变量，“Sbjct ”读入另一个标量变量。然后程序用下面的语句打开要读的文件：

```
open (IN, $blast_file) or die "$0: ERROR: $blast_file:
```

或如果由于某种原因找不到这个文件时就打印出错消息。然后，程序逐行检查文件，用 *chomp $line* ；去除换行符。最后使用打印功能，使程序读数据时显示当前行。

既然数据已在存储器中，需要将其分类，提取所需要的资料。记住，数据文件包含大量你想要忽略的多余资料。为此，要指示 Perl 仅仅考虑以 *Query* 或 *Sbjct* 开始的那些行。现在对 *Query* 和 *Sbjct*，除了想要的序列数据外，还有你不想要的不同长度的行号。为了正确读取序列数据，必须把程序设计成跳过行号，无论它们有多少字符，并且到达第一个核苷酸。你会注意到这行数据:

```
Query: 1165 gagcccaggagttcaagaccagcctgggtaacatgatgaaacctcgtctctac 1217
```

在 *Query*、开始行号、序列数据和结束行号之间有空格。因为恰好所有的 Query 和 Sbjct行都是这样，这成为如何正确读取数据的关键。为确定只获得所需的资料，把所有的query 和 subject 数据拆分成一个名为 @*words* 的四列数组。这个任务由下面几行程序完成：

```
    my @words = split /\s+/, $line;
    if ($line =~ /^Query/) {
     $query_src .= $words[2];
    } elsif ($line =~ /^Sbjct/) {
     $sbjct_src .= $words[2];
    }
}

# 已经读完 blast 文件，关闭
close IN;
```

一旦将数据读入 @*words* ,就命令程序只取@ *words* 的第二列数据（只有核酸序列数据），并将其存入变量 *$query_src*和 *$sbjct_src*。然后程序关闭文件并移到新的一行。现在你只是得到了以能利用的形式存储的数据。

程序的下一部分完成数据分析：

```
# 现在查找给定序列 ...
my @patterns =    gtccca    gcaatg    cagct    tcggga

# .. 找到后保存到散列中
my (%query_counts, %sbjct_counts);

# 搜索和存储序列
foreach my $pattern (@patterns) {
   while ( $query_src =~ /$pattern/g ) {
       $query_counts{ $pattern }++;
   }
   while ( $sbjct_src =~ /$pattern/g
       $sbjct_counts{ $pattern }++;
   }
```

程序中设置了一个运行五次的循环，每次搜索一个字符串或模式。在外部的*foreach*循环的每一次迭代中，程序运行内部的 *while* 循环，每次找到匹配模式增加一次计数。结果存储在独立的两个散列 *%query_counts* 和 *%sbjct_counts* 中。

这里是程序的最后一部分，生成输出数据:

```
# 创建空的报告文件
open (OUT, ">$REPORT_FILE"  ) or die "$0: ERROR: Can't write $REPORT_FILE";

# 输出文件的头部，包括
# 当前日期和时间
print OUT "Sequence Report\n",
    "Run by O'Reilly on ", scalar localtime, "\n",
    "\nNOTE: In the following reports, a dash (-) represents\n",
    "      missing data in the chromosomal sequence\n\n",
    "Total length of 'Query' sequence: ",
    length $query_src, " characters\n", "Results for 'Query':\n";

# 输出 Query 匹配
foreach my $key ( sort @patterns ) {
    print OUT "\t'$key' seen $query_counts{$key}\n";
}

print OUT "\nTotal length of 'Sbjct' sequence: ",
    length $sbjct_src, " characters\n", "Results for 'Sbjct':\n";

# 输出 Sbjct 匹配
foreach my $key ( sort @patterns ) {
    print OUT "\t'$key' seen $sbjct_counts{$key}\n";
}
```

```
close OUT;
__END__
```

这部分程序将编译并格式化结果，之后转储到文件 *report.txt*中。打开文件 *report.txt* 可以看到：

```
Sequence Report
Run by O'Reilly on Tue Jan  9 15:52:48 2001

NOTE: In the following reports, a dash (-) represents
      missing data in the chromosomal sequence

Total length of 'Query' sequence: 1115 characters
Results for 'Query':
        '-' seen 7
        'cagct' seen 11
        'gcaatg' seen 1
        'gtccca' seen 6
        'tcggga' seen 1

Total length of 'Sbjct' sequence: 5845 characters
Results for 'Sbjct':
        '-' seen 12
        'cagct' seen 2
        'gcaatg' seen 6
        'gtccca' seen 1
        'tcggga' seen 6
```

在这个例子中，结果送到文件中。也可以很容易地要求 Perl 创建一个可以用 Web 浏览器查看的 HTML 文件。或者也可以使程序能够进行人机对话，用 Perl 创建一个生成 Web 表单的 CGI 脚本来分析数据并返回结果。

我们现在只是初步了解了这种程序能做什么。还可以修改这个程序，使它从数据库中搜索更通用或更具体的模式。例如，可以搜索‘tcggga’和‘gcaatg’，但只有当它们由‘cagct’相连时才计数。也可以仅仅搜索数据中的间隔。完成所有的搜索后，可以用 Perl 自动地将所有结果存入一个数据库。

如果对此感到困惑，不必害怕。我们并不要求你理解所有的程序。正如我们在本章开头所说，这些程序并不是介绍如何使用 Perl 编程，而是展示 Perl 是如何工作的，以及编程并不是真的都那么难。如果有能力设计一个实验，那么你就有能力使用 Perl 或其他语言编程。

Perl 在生物信息学中的应用

值得庆幸的是，练习编程越多，能学到的东西就越多；而学到的越多，能做的事情就越多。用 Perl编写的程序都是有关分析数据和建立工具的。前面我们说过，生物学数据正在以令人惊异的速度激增。生物学家跟得上分析它们的惟一机会就是发展可重用软件工具的程序库。在 Perl 中，有大量能用于编程的可重用函数。把一组相关函数组装成模块（module），比把它们只用在一个完整程序中更好。在程序中，可以使用各种模块来实现它们支持的功能。还有一些其他科学学科以及游戏、图形程序设计、视频、人工智能、统计和音乐的 Perl模块。它们都是免费的。

为与其他 Perl 文件区分开，模块使用 *.pm* 作为后缀。要在 Perl 程序中使用一个模块的函数（假设是 CGI.pm），要在 shebang 行之后加入下面一行：

所有模块的原始来源都是 CPAN（Comprehensive Perl Archive Network ，综合Perl存档网络）。CPAN（*http://www.cpan.org* ）汇集了分布在世界各地的站点，它们都是在芬兰的 Perl主站点的镜像。想找到离你最近的 CPAN站点，请查询 Perl的网站（*http://www.perl.com*）。

由于有这么多可用的模块，在坐下来写一个新函数之前，值得花些时间来检查 CPAN 的档案，看看是否已经有人替你编写了这个函数。这一部分我们主要介绍一些对于解决生物信息学常见问题十分有效的 Perl 模块。所列出的并不全面，应该密切关注 CPAN 来了解最新的发展。

Bioperl

Bioperl Project（及其兄弟 Biopython、BioJava 和 Bioxml）致力于建立一个用于生物信息学研究的公开的源模块库。其总体思想是，把生物信息学中的普通数据项（例如序列和序列比对）描述成对象。这样，如果使用 Perl，不需要经常重写读写序列文件的程序，而只需要从Bio::SeqIO 调用适当的函数，把最后得到的序列数据转储到一个序列对象中。

Bioperl不仅限于存储序列：目前它还包括创建和存储序列比对、管理注解数据、从序列数据库搜索程序 BLAST 和 HMMer解析输出的模块，还有其他一些类似的模块。除了核心的 Bioperl 发布外，在 Bioperl 网站（*http://www. bioperl.org*）的 ScriptCentral还收集了许多与生物学相关的其他脚本。要学习更多关于 Bioperl 下载、安装、使用方面的知识，请访问 *http://www.bioperl.org*。

CGI.pm

CGI.pm 是一种用来设计交互式网页的模块。它的功能主要是格式化网页、创建和处理用户输入信息的表单。如果使用过 Web，几乎可以肯定你使用过用 CGI.pm 编写的网页。例如，创建一个使用 HTML 表单的页面，询问用户最喜欢的颜色是什么。当用户输入数据时，脚本将它存储在一个名为'color'的字段中。用户点击"Submit"时，相同的页面被装载，只有这一次，*$query->param('color')*包含这种颜色的名称，然后执行"else"之后的 *print*语句。CGI 脚本是下面的这种形式：

```
#!/usr/bin/perl

use CGI;                                 # 载入 Perl 的 CGI模块

my $query = new CGI;                     # 创建一个名为$query 的 CGI对象

                                         # 发送 HTML 首部和<HTML> 标签
print $query->header, $query->start_html;

                         # 如果用户首次访问网站，问他
                         # 最喜欢的颜色

unless ($query->param('color')) {        # 页面 1: 询问用户
    print $query->start_form, "What is your favorite color?
          $query->popup_menu( -name   => "color".
                    -values => ["red"  "green", "blue"] ),
          $query->submit,
          $query->end_form;
} else {                                 # 页面 2: 告诉用户
    print "Your favorite color is ", $query->param('color');
}

print $query->end_html;                  # 发送 </HTML> 标签
```

这个脚本的结果如图 12-1 所示。

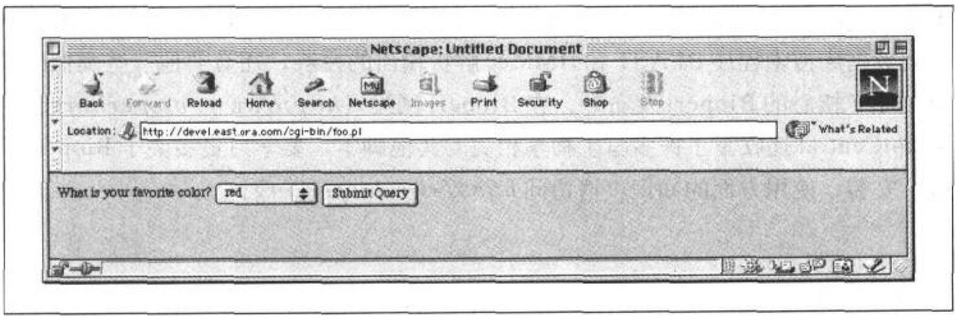


图 12-1　CGI 脚本生成了一个交互式网页

LWP

如果说CGI.pm 从服务器的角度使 Web 自动化，那么 Web 编程库（the Library for Web Programming, LWP）就是从客户机的角度使Web交互自动化。使用LWP，Perl 程序可以向表单提交数据、检索网页，并大大减少使用浏览器与 Web服务器进行人工交互时的单调和冗长。例如，假设想检索并打印出 *http://www.oreilly.com* 的主页的 HTML 源文件。可以使用 LWP::Simple 模块:

```
#!/usr/bin/perl
use LWP::Simple;
print get( "http://www.oreilly.com" );
```

这样来检索 *http://www.oreilly.com* 的 HTML 源代码，并在屏幕上显示。

PDL

Perl数据语言（Perl Data Language, PDL, 发音与"piddle"相同）是一种用矩阵进行数学运算的模块，常常用于科学应用和结合 GIMP 的图像处理（因为图像的计算机表示法恰好是矩阵）。在计算生物学中，PDL 在处理微阵列表达数据、得分矩阵以及来自图像的数据方面具有非常重要的作用。例如，在检测蛋白质 - 蛋白质相互作用的二维凝胶时，数据以图像形式存储，用图像处理技术就能对凝胶的特征参数进行定位和比较。

用 Perl 处理线性代数时并不需要一个完整的程序库。 PDL 允许处理任意维数的矩阵，与处理标量变量一样。例如，一个用标准 Perl 阵列构造的二维矩阵的形式为

$a[$i][$j]。如果想把两个基于阵列的矩阵（称之为 @a和 @b）相加并将结果存储在另一个矩阵 @c 中，就必须编写这样的程序：

```
for( $i=0; $i<$row_max; $i++ ) {
    for( $j=0; $j<$col_max; $j++ ) {
       $c[$i][$j] = $a[$i][$j] + $b[$i][$j];
    }
}
```

这样得写两个循环，外侧的一个迭代每一行，里面的一个迭代每一列。使用 PDL，只需简单地写成：

```
$c = $a + $b
```

换言之，把多维矩阵定义为 piddle（PDL 中矩阵数据对象的名称）而不是 Perl 数组时，PDL 使之在处理时如同简单的标量对象一样，即使是处理几兆字节的矩阵。此外还有用来测试 PDL 计算能力的叫做 *perldl* 的互动模式，这个互动模式与数字化数据分析软件包 R 和 Octave 提供的相似（我们将在第十四章中介绍）。

DBI

DBI（database interface，数据库接口的缩写）是一种用于编写关联数据库互动程序的模块。它允许编写一些程序，用来把数据放入数据库、查询数据库并从数据库中提取信息，而无需注意使用的特定数据库。例如，用 DBI 编写的脚本稍加改变就可与 MySQL 数据库或 Oracle 数据库一起使用。

GD

GD.pm 模块允许使用 Perl 程序生成图形。GD 常用来制作网页上简单的专用图表，例如 Web 服务器使用统计分析。PaintBlast.pm，是一个 GD 应用程序的例子，它是可以从 BLAST 输出的序列比对生成图像表示法的模块。可以从 Bioperl 的 ScriptCentral 得到该模块。

第十三章

构建生物学数据库

随着万维网（WWW）的建立和流行，生物学数据库已经成为生物学文献的重要组成部分。对一个科研工作者来说，学会如何从中央生物学数据知识库中找到、进入并下载信息与传统的生物学信息检索技能同等重要。主要在线资源，如PDB（Protein Data Bank）和GenBank，都是针对那些对数据库基本功能几乎没有了解的用户精心设计从而提供信息的，也允许那些不知道如何或者不想开发私人数据库的人向中心知识库存储数据。

然而，由于Web数据库在科学界内部共享信息变得越来越一体化，越来越多的人希望构建自己的数据库，并允许他们的同事直接登录。即使是像网站这样简单的东西也能被研究者改进，使得数据库的维护越来越容易。本章我们将介绍一些初步的数据库终端知识，举例说明如何构建简单数据集的数据库。

如果对计算机硬件和软件不是太熟悉，阅读本章并开发自己的数据库会比较困难。我们力图介绍构建数据库步骤的思路：建立数据模型、选择数据库管理系统（DBMS）、运行数据模型、给数据库设计一个友好的用户前端。本章的目的是让你对开发数据库出现的问题有一个总体了解，这将有助于你依靠自己或在数据库专家的帮助下解决问题。

使用数据库时并不需要了解数据库是怎样运转的。不过，在生物学界，提供一个通过Web到数据的入口变得越来越重要，为此，你必须至少具有关于数据库运转的最

基本的知识。即使你的实验室有足够多的钱来支付给你的 Oracle 管理员，你仍然需要知道一些知识。

数据库类型

有两种类型的数据库管理系统：平面文件索引系统和关系数据库管理系统。第三种类型，面向对象的数据库管理系统，开始越来越受欢迎。选择使用平面文件索引系统还是关系数据库系统对于数据库的计算效率和实用性具有深远的意义。

平面文件数据库

平面文件数据库是非专家也能理解的最简单的一种数据库类型。平面文件数据库并不是一个真正的数据库，而只是一个相似文件的有序集合，通常（而不是“总是”）是与其内容相符的标准格式。平面文件数据库强调格式化数据是就字符格式而言的，即打印时数据所呈现的格式。

平面文件的集合就好像一个塞满了纸的大箱子。平面文件数据库的按序排列和索引使得它较为有效。计算机文件系统上的平面文件集合可以排序并存储在带标记的文件夹中，就像是一堆打印出的纸张按顺序放在文件柜的抽屉中一样（见图 13-1）。在前面的章节中我们建议，利用文件系统的分级性质和切合实际的文件命名方案来使文件一目了然，实质上我们就是鼓励你自己建立一个初步的平面文件数据库。创建数据库意味着你可以记住数据库的规则，而不是某个文件的位置，并且可以更容易地解决问题。

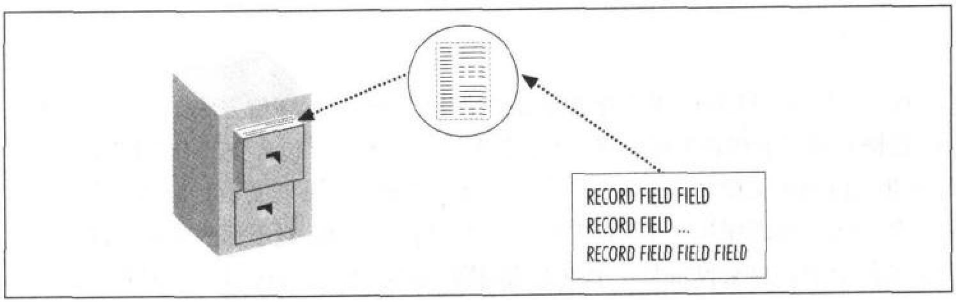


图 13-1　平面文件与平面文件数据库的关系

平面文件数据库常可通过索引进行搜索。索引从文件中提出其特别的属性，在索引中把属性值与文件名、位置配对。这就像是一本书的索引，例如，它告诉你在书中何处会找到‘genome”这个词。与书的索引一样，数据库索引需要仔细设计，以使索引只有当某个词出现在能提供资料的上下文中时才指向该词。数据库索引通过分别给文件的不同字段编索引而标识出上下文。“cytochrome”一词出现在蛋白质结构文件的 Molecule Name 字段中对于使用者大概比只是出现在文件注释中重要得多。在第一种情况下，找到“cytochrome”一词就能确定文献中包含有关某些细胞色素分子的信息。而在第二种情况下，这个词可以出现在文章标题或是一个关于分子间相互作用的注释中，即使文件中细胞色素这个结构实际上属于另一个不同的分子。如果创建多重索引，那就可以根据关键字搜索一个特定的索引文件，这比在数据库中逐个搜索实际文件要简便得多。

平面文件数据库没有内部缺陷。它们以切合实际的方式组织数据，而且采用正确的索引，可以对它们进行大范围的搜索。不过，当平面文件集合越来越大时，对它们的操作效率就降低了。索引是一维的，因此很难（而非不可能）在被索引的平面文件数据库内把不同属性相连。

生物学中的平面文件数据库

许多常用的生物学数据库开始就是平面文件数据库，这也就导致了许多我们在前面的章节中讨论过的程序和软件包对于输入数据行格式有严格的内在要求。

例如，PDB从使用著名的PDB格式的平面文件开始。这些平面文件的格式被设计成易于用 FORTRAN程序读取，而且实际上当计算机在穿孔卡上输入数据编码时就已是这样了。当只有几个蛋白质结构文件时，维护和访问这个数据库毫无问题。直到1990年，PDB产生将近20年后，它只是发展到几百个文件。

20世纪90年代，PDB的快速发展导致需要找到新的数据存储方法。在实际应用方面，数据库的完全列表开始变得很长，如果用户进入一个包含全部有效PDB文件的目录并试图列出文件名，即使是产生一个文件清单也需要花费数秒时间。读取庞大目录的内容使得即便*ls*这样简单的Unix工具也慢了下来，对于那些要重复读取一个目录的计算机程序来说这是一个更大的问题。起先，PDB根据PDB代码的字母分成子目录。但当数据库达到将近8000个条目时，开始表明这个方法也非常麻烦。

PDB 现在使用面向对象的数据库后端（这部分运算驻留在 PDB服务器上，用户看不到）来支持数据库查询和文件存取。然而，有效文件格式仍继承了 PDB的格式，以便用户仍能使用那些 PDB 改进之前所开发的软件。

除了 PDB 之外，平面文件数据库还广泛地应用在生物学中。许多生物序列数据的使用者使用 SRS（Sequence Retrieval System，序列检索系统）在本地存储和访问序列，这是一种为生物学数据设计的平面文件索引系统。

关系数据库

和平面文件数据库一样，关系数据库只是汇集关于某个事物的所有信息并将其存储在计算机中的一种方式。在平面文件数据库中，某个事物的所有信息都存储在一个大的高度结构化的文本文件中。而在关系数据库中，这些信息存储在一个表格的集合中。

描述蛋白质结构的平面文件像是一本精装书。包括关于这个样品的来源、如何收集数据、序列、二级结构和原子的位置等各个章节。

在关系数据库中，每个章节中的信息被放进单独的表格中，每种蛋白质拥有自己的一组表格，而不是一本书。因此就有一些试验条件、二级结构元素、原子位置等的表格。所有这些表格以它们所描述的蛋白质做标记，这样就可以把它们相互联系，但并不像一本书那样结合在一起。数据库中的表格格式都遵循一致的规则，因而你可以立即读取所有关于原子位置的表格或所有关于试验条件的章节，就像读取有关一个特定蛋白质的所有表格一样容易。

如果只是对某一个特定蛋白质感兴趣，你可以非常方便地去图书馆（PDB），在目录中查找这本书，然后通读。图书管理员可以从书中挑出几条信息（例如蛋白质名称、最先沉淀出这种蛋白质的发现者等）并放在一个索引中（像一个卡片目录），这可以帮助你找到这本书在书架上的位置。

但如果你想获得蛋白质图书馆的每一本书中关于二级结构的章节又该怎样呢？你不得不到图书馆，从书架上取下每一本书，复印二级结构一章，然后把那些信息转换成易于分析的形式。

RDBMS（关系数据库管理系统）允许把数据库中的所有蛋白质结构数据视作整体。你可以从许多不同“角度”来“看”这个数据库，并且只抽出需要的信息而无需把每本书中特定的章节复印出来。由于关于蛋白质的每一条独立的信息都存储在数据库的独立的表格中，RDBMS 可以飞快组装出你想要的任何一种关于蛋白质的“书籍”。如果你想要一本关于血色素的书，没问题。更好的是，RDBMS很容易给你制作出一本关于数据库中所有蛋白质二级结构的书。

你所要做的只是想出如何建立恰当的查询以从数据库返回你想要的信息。如果你想要一本关于血色素的书，你可以告诉 RDBMS “if protein name equals hemoglobin then give me all information about this protein” （如果蛋白质名称是血红素，就给我所有关于这个蛋白质的信息）。如果你只想要数据库中描述血色素二级结构的每个条目，你可以告诉 RDBMS “if protein name equals hemoglobin then give me the secondary structure table about this protein” （如果蛋白质名称是血红素，就给我这个蛋白质二级结构的表格）。

如何编制表格

关系数据库表格中的数据按行编制，每一行代表数据库中的一条记录。一行可以包含几条独立的信息（字段）。数据库中的每个字段必须包含一条单独的信息，而不能由可被进一步拆分的组或列表组成。

关系数据库中的表格并不仅仅是美化的平面文件，尽管如果打印出来，它们看起来是这样。行（row）是记录（record）的同义词，而不是指 80个字符的排列。每一行中的字段不受字符数的限制，字段在其意义完结之处结束。 RDBMS的工作是通过快速查找共同元素把相关表格联系起来。

通过比较 PDB中两种类型的蛋白质结构数据文件，可以对组织成表格的数据和字符格式的平面文件数据之间的区别有一个认识。标准 PDB文件被排列成一系列 80字符的行。每行做一个标记，但标记关联的信息十分不均匀，尤其是在标题中。例如：

```
REMARK  1                                                      4HHB  14
REMARK  1 REFERENCE 1                                          4HHB  15
REMARK  1  AUTH   M.F.PERUTZ,S.S.HASNAIN,P.J.DUKE,J.L.SESSLER,  4HHB  16
REMARK  1  AUTH 2 J.E.HAHN                                     4HHB  17
REMARK  1  TITL   STEREOCHEMISTRY OF IRON IN DEOXYHAEMOGLOBIN  4HHB  18
REMARK  1  REF    NATURE                  V. 295   535 1982 4HHB  19 REMARK
1  REFN    ASTM NATUAS  UK ISSN 0028-0836            006  4HHB  20
```

```
REMARK   1 REFERENCE 2                                                4HHB  21
REMARK   1  AUTH    G.FERMI,M.F.PERUTZ                                4HHB  22
REMARK   1  REF     HAEMOGLOBIN AND MYOGLOBIN.    V.   2         1981 4HHB  23
REMARK   1  REF   2 ATLAS OF MOLECULAR                                4HHB  24
REMARK   1  REF   3 STRUCTURES IN BIOLOGY                             4HHB  25
REMARK   1  PUBL    OXFORD UNIVERSITY PRESS                           4HHB  26
REMARK   1  REFN              ISBN 0-19-854706-4                 986  4HHB  27
```

在这个 PDB 引用记录中，可以看到，每一行中的条目并不是单独的信息，每一行也不均匀。有时在一行中有四个发现者的名字，有时有两个。有时有三个标题线，有时只有一个。这会导致分析或用计算机程序读取标题时的困难。

把这个与 mmCIF 文件比较。mmCIF 是一种 X 射线晶体学试验结果采用的新的数据标准。由于在 1999 年改变了 PDB 的管理程序，可以从 PDB 得到 mmCIF格式的蛋白质结构。

在mmCIF文件中数据之前的部分看起来很像是一连串计算机程序中的命令，这些行说明了如何读取文件中的数据。然后就是数据表格。例如：

```
loop_
_citation.id
_citation.coordinate_linkage
_citation.title
_citation.country
_citation.journal_abbrev
_citation.journal_volume
_citation.journal_issue
_citation.page_first
_citation.year
_citation.journal_id_ASTM
_citation.journal_id_ISSN
_citation.journal_id_CSD
_citation.book_title
_citation.book_publisher
_citation.book_id_ISBN
_citation.details

primary   yes
; THE CRYSTAL STRUCTURE OF HUMAN DEOXYHAEMOGLOBIN AT
    1.74 ANGSTROMS RESOLUTION
;
UK 'J.MOL.BIOL.                    '  175  ?     159   1984
'JMOBAK          ' '0022-2836                  '  070 ? ? ? ?

    1       no
; STEREOCHEMISTRY OF IRON IN DEOXYHAEMOGLOBIN
;
```

```
UK 'NATURE                        '  295  ?     535   1982
'NATUAS          ' '0028-0836                  '  006 ? ? ? ?

2        no
?   ? ?    2 ?     ? 1981 ? ?  986
; HAEMOGLOBIN AND MYOGLOBIN.
  ATLAS OF MOLECULAR
  STRUCTURES IN BIOLOGY
;
;  OXFORD UNIVERSITY PRESS
;
'0-19-854706-4           ' ?
```

mmCIF 文件中包含的是许多有关同一蛋白质的表格。

mmCIF 文件中参考(reference) 部分的开头几行（就是完全描述一个蛋白质结构的表格集合的平面表示法）说明了将要出现的表格每一行中的字段的意义。行并不是随意地从第 1 个字符开始，到第 80 个字符结束，从数据的打印输出或屏幕显示来看，它们可能延伸成几行。行直到它们的所有字段都已被填入时才结束；信息缺失时（像前面的例子），字段中填入空字符，如问号或空格。

在蛋白质数据库中，描述一个特定结构的参考文献表格与一个特定的 PDB ID 相关联。不过，还有其他的表格也与那个 PDB ID 相关联，而且它们有与参考表格完全不同种类的行。描述蛋白质结构的原子位置包含在一个格式完全不同的独立表格中：

```
loop_
_atom_site.label_seq_id
_atom_site.group_PDB
_atom_site.type_symbol
_atom_site.label_atom_id
_atom_site.label_comp_id
_atom_site.label_asym_id
_atom_site.auth_seq_id
_atom_site.label_alt_id
_atom_site.cartn_x
_atom_site.cartn_y
_atom_site.cartn_z
_atom_site.occupancy
_atom_site.B_iso_or_equiv
_atom_site.footnote_id
_atom_site.label_entity_id
_atom_site.id
1
 ATOM  N  N    VAL A   1  .   6.204  16.869   4.854  7.00 49.05  .    1  1 1
 ATOM  C  CA   VAL A   1  .   6.913  17.759   4.607  6.00 43.14  .    1  2 1
 ATOM  C  C    VAL A   1  .   8.504  17.378   4.797  6.00 24.80  .    1  3 1
 ATOM  O  O    VAL A   1  .   8.805  17.011   5.943  8.00 37.68  .    1  4 1
```

```
ATOM  C  CB   VAL A   1  .   6.369  19.044   5.810  6.00 72.12  .    1  5 1
ATOM  C  CG1  VAL A   1  .   7.009  20.127   5.418  6.00 61.79  .    1  6 1
ATOM  C  CG2  VAL A   1  .   5.246  18.533   5.681  6.00 80.12  .    1  7 2
ATOM  N  N    LEU A   2  .   9.096  18.040   3.857  7.00 26.44  .    1  8 2
ATOM  C  CA   LEU A   2  .  10.600  17.889   4.283  6.00 26.32  .    1  9 2
ATOM  C  C    LEU A   2  .  11.265  19.184   5.297  6.00 32.96  .    1 10 2
ATOM  O  O    LEU A   2  .  10.813  20.177   4.647  8.00 31.90  .    1 11 2
ATOM  C  CB   LEU A   2  .  11.099  18.007   2.815  6.00 29.23  .    1 12 2
ATOM  C  CG   LEU A   2  .  11.322  16.956   1.934  6.00 37.71  .    1 13 2
ATOM  C  CD1  LEU A   2  .  11.468  15.596   2.337  6.00 39.10  .    1 14 2
ATOM  C  CD2  LEU A   2  .  11.423  17.268    .300  6.00 37.47  .    1 15
```

原子(atom)表格中的值明显与参考表格的值相关(related);它们都含有关于相同PDB结构的信息。然而不能只是这两种类型的数据一起放进一个大表格中。无论是通过在表格“底部”添加还是通过添加额外的列添加(尽管在平面文件中恰恰要求如此操作),把参考信息放进原子信息所在表格的行和列中是没有意义的。这两种数据类型是相关的,但相互正交(orthogonal)。

在一组信息中任何无法合理地把行或列添加到一个表格中的地方,都需要创建新的表格(注1)。数据库中的表格可能只是在最高层次才有关系,例如原子和参考信息与同一个PDB文件相关,或者它们更紧密地链接在一起。

你可能在前两页的参考记录中注意到发现者的名字并未列出。这是怎么回事呢?答案就是它们在一个独立的表格中。因为每一个参考都可以包含任意数量的独立的作者,这个信息不能仅仅通过增加固定数量的行或列而添加到参考表格中。因而有一个作者姓名的独立表格:

```
loop_
_citation_author.citation_id
_citation_author.name
 primary   'Fermi, G.'
 primary   'Perutz, M.F.'
 primary   'Shaanan, B.'
 primary   'Fourme, R.'
   1       'Perutz, M.F.'
   1       'Hasnain, S.S.'
   1       'Duke, P.J.'
   1       'Sessler, J.L.'
   1        'Hahn, J.E.'
   2       'Fermi, G.'
```

注1: 将复合数据集分成一组正交相关表格的过程称为规范化(normalization)。关系数据库理论的精确论述参见参考文献中的相关文献。

```
2        'Perutz, M.F.'
3        'Perutz, M.F.'
4        'TenEyck, L.F.'
4        'Arnone, A.'
5        'Fermi, G.'
6        'Muirhead, H.'
6        'Greer, J.'
```

这个表格与前面的参考表格通过第 1列的值相关联，它与其他参考表格中的引文ID相对应。要在数据库中根据“ Fermi, G.”找到‘ THE CRYSTAL STRUCTURE OF HUMAN DEOXYHAEMOGLOBIN AT 1.74 ANGSTROMS RESOLUTION”,就要通过引文 ID，它确定了这两个实体之间的关系。

使用 RDBMS 开始看上去好像是过多考虑了一组非常简单数据的存储问题。如果你编写过基于过时平面文件PDB格式的程序，就会认识到它在给关系数据库中的表格赋值时是多么有用。其中，数据库减少了复杂的行格式语句和采用 80字符格式文件时所需要的解析操作。

数据库模式

组成数据库的图表网络以及它们之间的相互关系称为数据库模式（ database schema)。要长时间地保持数据库的实用性，最好在考虑使数据库流行之前就谨慎地改良其模式。在本章以后的例子中， 我们将开发一个简单数据库的模式。

不提出你自己的模式而想要理解数据库模式和表格是很困难的。不过，关系数据库是大型数据库运算的标准，理解其概念对于任何一个想建立自己的关系数据库的人来说都是非常必要的。开始设计自己的数据库之前，应该查阅严格论述关系数据库的参考文献。

面向对象的数据库

你可能听说过与编程语言和数据库有关的“面向对象（ object-oriented)”这个词。面向对象的数据库系统是一种与面向对象的编程规则一致的数据库管理系统。面向对象的数据库有一些重要的性质：它们是为解决多用户并行互动而设计的；能处理复杂的对象（字符数据表除外）；具有持续性(persistent)，也就是说，能保留执行

的过程。由于面向对象的编程策略在实际操作过程中很流行，大多数主要关系数据库管理系统在一定程度上都能和面向对象的方法相兼容。

数据库中面向对象方法的实际结果就是DBMS的出现，它的功能不仅仅局限于存储表格和处理用严格定义的查询语言词汇写成的函数。由于面向对象的数据库把数据作为对象而不是表格来处理，它可以为同一个数据库中从简单的文本格式数据到图像和视频文件的每一个数据提供入口。面向对象的数据库不强制使用 SQL 查询语言，而是为编程语言提供了灵活的结合点。许多 DBMS 已开始同时具有对象和相关的特征，但占主导地位的仍是相关 DBMS。

数据库软件

数据库建立后要靠DBMS来维护。有几种DBMS，有些是公用资源，有些是开源的，有些是商品化的，包括平面文件索引系统、 RDBMS、ODBMS（对象数据库管理系统）以及对象 －关系混合的数据库管理系统。采用何种DBMS 取决于你所提供的数据、对软件的熟悉程度以及目的。

SRS

如果你已经决定采用平面文件索引和检索系统，你也不必再从头开始这个工作。SRS（Sequence Retrieval System，序列检索系统）是一种常用的平面文件管理系统，广泛应用在生物学领域，包括商业和学术两方面。 SRS 是EMBL 特别为在分子生物学数据库中的应用而开发的，现在可以从 Lion Bioscience（*http://www.lionbioscience.com*）以商品形式得到。它仍可连同大量的资料一起（但没有技术支持）向学术机构的研究人员免费提供。 SRS 数据库的一个最常见的应用就是维护主要生物学序列数据库的本地镜像。当前版本是 SRS 6。

SRS 可以在 SGI、Sun、Compaq和 Intel Linux 系统安装。要维护你自己的 SRS数据库和主要生物学数据库镜像需要几十吉的硬盘空间，因此这并不是能轻松做成的事。SRS有内置的解析程序，知道如何读取EMBL核苷酸数据库文件、SWISS-PROT文件和 TrEMBL 文件。还可以使用 SRS的 Icarus语言开发数据库模块把其他数据库整合到 SRS中。如果要找一个在SRS平面文件管理系统中可以整合的这类数据库的

例子，只需查看 SDSC Biology Workbench即可。直到最新的版本，SRS 都是在生物学工作台中使用的数据库管理系统，几乎支持现在汇集到工作台中的所有数据库。

Oracle

Oracle是 RDBMS 中的领头羊。它是按照行业标准设计的商品化软件，具有非常大的容量。它也正迅速成为联邦资助研究项目的标准。 Oracle 具有一些对象容量（object capacity ）以及大量的关系容量（ relational capacity）。对 Oracle 感兴趣的用户现在可以从 *http://www.oracle.com* 获得免费试用 Oracle 的许可。如果想向生物学界提供大规模的数据资源，可能就需要使用 Oracle 的开发工具。

PostgreSQL

PostgreSQL 是一种具有对象 - 关系全部特征的 DBMS ,除大量 SQL 函数和数据类型外，它还支持用户定义的数据类型和函数。 PostgreSQL 是一个源代码开放的软件，可以从 *http://www.postgresql.org* 免费下载其源代码，这个网站还在线提供大量的 DBMS 资料。PostgreSQL 还可以在大多数标准Linux 产品中找到。如果想创建一个包含特殊类型数据的数据库，而且很有可能对将来的数据库扩充， PostgreSQL 可能比 MySQL 更能满足需要。相对于 Oracle和其他商业 DBMS而言， PostgreSQL 在处理大量运算的效率方面略显有限，但对于中型数据库，它是一个优秀的产品。

源代码开放的 ODBMS

开发源代码开放的 ODBMS 现在刚刚起步。其中最引人注目的是 Ozone Project（*http://www.ozone-db.org*）。Ozone全部是用 Java 完成的，而且是为 Java程序员设计的，查询是用底层语言而不是 SQL（Structured Query Language，结构化查询语言）实现的。Ozone 开发的一个重点就是对象持续性（ object persistence），即DBMS 保持数据对象状态而不受使用者处理影响的能力。和许多 ODBMS 一样，Ozone 还在开发的初级阶段，对于一个新的使用者来说并不特别容易理解。在对象技术成熟之前，开发数据库时比较明智的可能是坚持采用关系数据库模块，除非你有令人信服的理由采用面向对象的规则。

MySQL

MySQL 是一种源代码开放的关系数据库管理系统，其建立和使用相对简单，而且Unix和 Windows操作系统都可以使用。MySQL的特征丰富而复杂，它和其他两种常用的 RDBMS PostgreSQL和 Oracle稍有不同。每个系统识别一个不同的 SQL数据类型和函数子集，没有哪一个能识别所有可能的类型。 MySQL 设置了比PostgreSQL和 Oracle低的容许运算数目极限，因此有时认为它适用于小型和中型数据库，而不是大型数据库项目。然而并没有严格的规定，这取决于你想对数据库中的数据进行什么运算。MySQL是一种严格的 RDBMS ，因此，如果你想存储特殊类型数据，这可能并不是合适的DBMS。不过，对于大多数标准数据库应用程序而言，MySQL 是一个极好的起点。

MySQL的开发者声称它可以比其他 DBMS更迅速地管理大型数据库。他们的标准测试程序证明了这一点，不过我们没有对其进行过独立的评估。我们能说的就是，学会使用MySQL，建立一个基本的数据库并在几小时到几天内运行，这些是完全可能的，这取决于使用者使用 Unix 和 SQL 的经验。

SQL 概论

在实际工作中，最有可能使用的是 SRS这样的生物学数据的专业平面文件数据库系统和某些 RDBMS。为了使用相关数据库管理系统，需要对 SQL有所了解。

SQL，即结构化查询语言（Structured Query Language，了解它的人发音为“see-kwl”，拘泥于字面的人发音为“ess-que-ell”，爱开玩笑的人发音为“squeal”）是RDBMS使用的语言。SQL命令在 DBMS接口中发出，而不在 Unix命令行发出。你向 DBMS程序发出命令，再由程序解释。

SQL命令可通过其他程序（例如，从 Web表单收集数据的脚本）或手工输入传递给DBMS。显然，第一种方式比较理想，尤其是对于向数据库输入大量记录而言，我们并不想手工操作。不过我们无法告诉你用 SQL编程的所有细节，本节我们只是着眼于介绍基本 SQL 命令及其功能。稍后我们将举一个可与 SQL 数据库互动的基于Web的程序的例子。

SQL命令读起来像不自然的英语，词汇非常有限。如果你记得高中英语课中的图解句子，找出主语－动词－宾语关系和条件从句，SQL就显得相当直观了。难题是记住词汇和语法的限定，以及构建查询以使DBMS能理解它们。SQL语句可能像下面这样读取数据（注 2）:

```
SELECT program FROM software WHERE program LIKE 'blast'
```

意思是"select the name of programs from the list of software where the name of the program is like blast"。如果使用一个生物信息学软件的可搜索数据库，这可能就是你想做的。

前面提到过，所有的DBMS并不相同。有一个SQL标准词汇表，称为SQL92。不过大多数系统只是实现了部分标准。你需要学习所使用的具体DBMS的文档,这样就不会因为发出了未被识别的命令而发生混乱。

SQL数据类型

数据类型的概念很容易理解。数据类型（datatype）是一个描述存储在表格特定列中的数据的形容词。通常，存储在表格中的数据可由数值或字符串组成。不过，SQL在这些常见数据类型中定义了大量的子类型，多数变量对于文本字段或数字字段的大小设定了不同的上限，还定义了一些特殊数字类型，如DATE和MONEY。

在数据库中创建表格时，需要定义每一列的类型。这意味着在建立数据模型时，从一开始就要知道每一列中包含的是何种类型的数据。还应该对数据可能的上、下限有大致的了解，这样就能选择包含它们的最小的可能类型。例如，如果知道表格的一列中的整数值不会大于255，这一列应该使用最小的可能整数类型TINYINT，而不是在实际上并不需要的时候在数据库中为大得多的数值让出空间。另一方面，如果数值最终会超过255，这一列就应该选择允许更大范围数值的类型。建立关系数据库是需要深思熟虑的。

注 2: SQL命令不需要全部用大写字母，它们是不分大小写的。但在我们的例子中它们都用大写字母，以便很容易地把它们从文件、变量和数据库名称中区分出来。SQL中文件、变量和数据库的名称是区分大小写的，因此如果你把一个数据库命名为PeOPlE，就必须保持这个形式。

下面是一些最常用的 SQL 类型，它们都被大多数主要的 RDBMS 程序支持：

INT

整数。变化类型包括：TINYINT、SMALLINT、MEDIUMINT 和 BIGINT，它们允许不同范围的数值。

FLOAT

浮点数字。最大值约为 3 E 38，最小值约为 1.7 E-38。

REAL

更长的浮点数字。最大值约为 2 E 308，最小值约为 2 E-308。

CHAR

定长文本字符串。值化填入固定长度的空格短。

TEXT

具有最大值的可变长度文本字符串。其变化类型包括 TINYTEXT、MEDIUMTEXT 和 LONGTEXT。

BLOB

具有最大值的可变长度二进制字段。变化类型包括 TINYBLOB、MEDIUMBLOB 和 LONGBLOB。是能放进二进制字段的任何数据。LONGBLOB 的最大值是 4GB。各种数据，如图像数据，均可放进二进制字段。

DECIMAL

以字符串形式而不是数字字段存储的实数。

DATE

存储年、月、日的日期值。

TIMESTAMP

每当记录改变时更新的时间值。

ENUM

一组限定选项中的一个值，并可以通过选项名称或代表名称的数值来选择。

SET

一组限定选项中的一个值。

SQL 命令

SQL 有许多命令，但现在对我们来说最重要的可能是了解怎样创建新的表格、怎样向其中添加数据以及怎样在数据库中搜索数据。我们将简要介绍 SQL 的CREATE、ALTER、INSERT、UPDATE 和 SELECT 命令，以在 MySQL 中实现为例。参考文献中提到的参考资料包含在 MySQL 中用得到的 SQL 命令的全部说明。

向数据库添加表格

使用 SQL CREATE 语句创建新表格。CREATE 语句的语法很简单：

```
CREATE TABLE tablename (columnname type [modifiers] columnname type
[modifiers])
```

如果想创建一个软件包信息的表格，以本章中我们讨论的数据库为例，可以像下面这样做：

```
CREATE TABLE software_package
       (packid INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
        packname VARCHAR(100)
        packurl VARCHAR(255)
        function TEXT
        keyword ENUM
        os SET
        format SET
        archfile VARCHAR(255)
```

这个命令告诉 MySQL 建立一个表格，其中第一列是自动增加的整数，即 DBMS 自动给建立的每一个条目赋一个惟一值。第二列和第三列是预置最大长度的可变长度字符串，在其中将输入软件包的名称和 URL。第四列是文本字段，可以包含描述软件包的 64KB 以下的文本。第五列可以从 64 个预置关键字中选择一个描述软件包。第六列和第七列可以从一组预置值中选择任何一个值来描述软件运行的操作系统（如 *mac*、*windows*、*linux*）和档案文件有效的类型(如 *binary*、*rpm*、*source*、*tar*)。最后一个字段是另一个可变长度字符串，含有档案文件的 URL。

修改已有表格

如果创建一个表格，而且想让它与原来设计的看上去有所不同，可以使用 ALTER 命令修改。要向表格中加入另一列，语法是：

```
ALTER TABLE tablename ADD [COLUMN] (columnname type [modifiers])
```

向已有表格添加数据

INSERT 命令可以向表格中加入新的一行数据。 INSERT 命令的语法是：

```
INSERT INTO table ( colname1, colname2, colname3 ) VALUE (
'value1', 'value2', 'value3')
```

修改表格中的已有数据

UPDATE 和 REPLACE 命令可以修改表格中已有的行。你的用户特权必定允许你使用 UPDATE 和 REPLACE。这些命令可以采用 WHERE 子句，语法和 SELECT 命令相似，因而可以指定在什么情况下更新记录。

用 SQL 的 SELECT 命令访问数据库

SQL 的 SELECT 命令可以在表格中查找数据。换言之，SELECT 是一个使数据库创建并填充数据后有效的命令。可以用一个让你决定在什么条件下从表格中选择一条记录的条件子句修改。

挑选要选择的字段

SELECT 命令的通用语法是：

```
SELECT [fields] FROM [table] WHERE [clause]
```

要选择特定表格中的全部字段，可以使用通配符 *：

```
SELECT * FROM [table] WHERE [clause]
```

在本章的数据库例子中，如果想从软件表格中选择软件包名称和 URL，SELECT 命令是：

```
SELECT packname, packurl FROM software
```

使用 WHERE 子句指定选择条件

用 WHERE子句可以指定在何种条件下可以从表格中选择记录。可以使用标准运算符，如 =、>=等，来为 WHERE 子句设定条件。MySQL 也允许在模式匹配时使用 LIKE和 NOT LIKE 运算符。

如果想建立 SELECT 语句查找序列比对的软件，应写成这样：

```
SELECT packname, packurl FROM software WHERE keyword = "sequence alignment";
```

如果想找到名称以字母 B 开始的软件包，SELECT 语句如下所示：

```
SELECT packname, packurl FROM software WHERE packname LIKE "B%"
```

% 符号是一个通配符，表示任意数量的字符，因此用这个语句选择的软件包的名称可以是任意长度，只要是以 B 开始。

多个表格联合输出数据

SELECT还可以连接两个相关的表格。后面我们谈到开发数据库时，你会发现表格之间的关系是通过从一个表格复制主键（primary key）信息到另一个表格中作为外键（foreign key）的方法建立的。如果一个表格中的外键与另一个表格中的主键匹配，两个表格中的数据指向相同记录，而且可以连接在一起产生一组来自 SELECT 的输出数据。连接两个表格的 MySQL SELECT 语句可能是这样：

```
SELECT FROM table1, table2 WHERE primarykey1=foreignkey2
```

例如，我们已经讨论了创建一个列出数据库中软件包的名称、 URL和其他细节的表格。为了正确地建立数据库，必须有另外一个表格，列出描述数据库中的软件包的功能的参考资料。

如果只想选择那些在 1998 年或更晚才第一次在文献中描述的软件包的名称和 URL 又该怎样做呢？名称和 URL是在 software 表格中找到的，而日期则在 reference 表格中，SQL 语句如下：

```
SELECT packname, packurl, reference_date FROM software, reference
WHERE software.package_id = reference.package_id
AND reference_date >= "1998";
```

变量 *package_id*是来自 software表格的主键，它在 reference 表格中复制以维持用来描述 software package （软件包）和 reference （参考资料）的细节之间的关系。如果*package_id*在两个表格中的值相同，被访问的这两行就是数据库中相同记录的一部分。因此，WHERE子句的第一部分连接了两个表格。 WHERE 子句的第二部分（*AND reference_data >= "1998"*）指定实际的选择条件。

不同的数据库管理系统实现不同水平的连接功能，因此必须查阅 DBMS文档，看连接是如何工作的。

安装 MySQL DBMS

要建立和维护自己的数据库，需要在存储数据的机器上安装一个数据库服务器。

MySQL 是一种安装和运行都非常简单的小型关系 DBMS 。我们将用 MySQL 建立范例数据库，因此如果你有兴趣尝试一下，请确定机器中已经安装了 MySQL 服务器。如果使用的是 Red Hat Linux产品，这就极其简单了。如果机器中没有安装 MySQL，只要使用 *kpackage*或 *gnorpm* 来选择你想安装的 MySQL 组件 —— 服务器、客户程序和开发工具就可以了。这样给你的是较早版本的 MySQL ，要得到当前版本并轻松地安装，请使用 *http://www.mysql.com* 上最新稳定版本的二进制RPM（Unix 软件产品）。还要确定已安装了 Apache Web 服务器和 PHP 支持程序（可以从 *http://www.apache.org*得到）。安装后重新启动计算机，MySQL服务器的守护进程（注 3）*mysqld* 启动，MySQL特许数据库初始化，PHP 模块对 Apache 服务器而言可用。

安装 MySQL

从RDBMS 软件中通常会发现需要选择安装客户程序还是服务器。程序的服务器部分是在实际存储数据的机器上运行。在Unix 计算机上它作为守护进程运行，即作为一个始终打开的系统进程从客户程序接收请求并做出反应。 MySQL 服务器称为 *mysqld* 。图 13-2 所示为一个客户程序／服务器结构的例子。

注 3: 在 Unix 系统后台运行的服务器等系统进程称为守护进程（ daemon）。

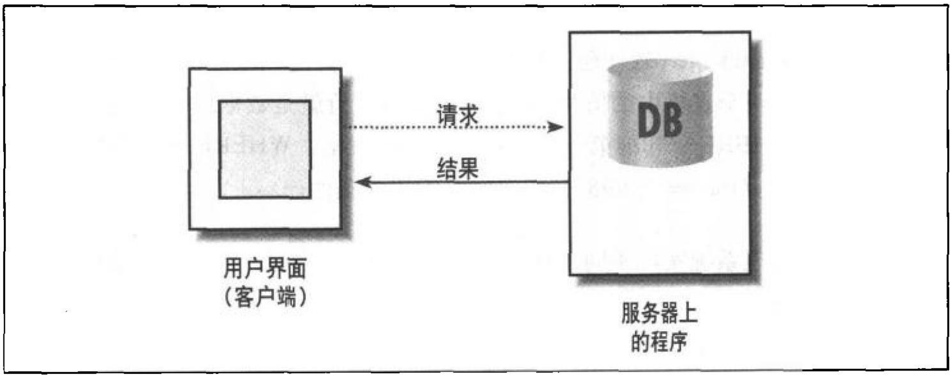


图 13-2　客户程序／服务器结构

客户程序(client)是连接服务器和请求数据的程序。客户程序可以在数据库服务器上，也可以在未运行 *mysqld* 而通过因特网连接到主数据库的其他计算机上。

MySQL 客户程序包括：*mysql*，主要的客户程序，可用来完成从设计数据库到插入记录的各种工作；*mysqladmin*，完成选定的管理功能，例如创建新的数据库和检查服务器的状态；*client5*，在允许交互式查询处理方面与 *mysql* 相似，但为安全起见，它不允许添加或修改数据库记录。

把 MySQL DBMS 作为整体讨论时，我们用 MySQL 表示。讨论作为 MySQL 的一部分的客户程序时，我们用具体的客户程序名称表示。

使用 mysql 客户程序

mysql 程序只有几个它自己的命令，主要指向 *mysql* 程序或 *client5* 程序的命令是 SQL 语句。进入 *mysql* 程序后，程序把你给它的任何 SQL 语句都作为一个连续的语句解释，直至终结字符“；”出现。下面是 *mysql* 命令：

use

　　把数据库名称作为参数；允许改变正在使用的数据库

status

　　返回服务器的状态

connect

与服务器重新连接

go 向 MySQL 服务器发送命令，也可用以“\g”或“;”终止 SQL 语句的方式来表示

help

打印 *mysql* 命令一览表

使用 mysqladmin 客户程序安装 MySQL

可以用下面的命令得到 *mysqladmin* 命令一览表：

```
mysqladmin --help
```

下面是一些最常用的命令：

create

以数据库名称作为参数；创建新数据库

drop

以数据库名称作为参数；删除整个数据库

reload

重新加载授权表格

variables

打印描述 MySQL 安装的有效参数

ping

检查 MySQL 服务器是否有效

shutdown

在本地计算机关闭 MySQL 服务器

重新启动 MySQL 服务器

mysqladmin 有一个关闭服务器的选项。但怎样重新启动它呢？要启动 MySQL 服务

器，不管是用户 *mysql* 还是其他用户 ID，都可以使用 Unix 的 *su* 命令成为 MySQL 管理员，然后用下面的命令启动 MySQL服务器：

```
safe_mysqld
```

MySQL服务器的安全保护

第一次从 RPM安装时，尽管数据库已初始化，但MySQL 服务器并不安全。要保护服务器，应立即设置 MySQL 安装的根口令。这可以（而且应该）与系统根口令不同。MySQL用户名和系统用户名之间没有关系，尽管服务器进程要在服务器的用户ID之下运行。必须直接使用 *mysql* 程序更新用户授权(user grant)表格，这是 MySQL用户权限的主要表格。用下面的命令调用 *mysql* 程序：

```
mysql -u root mysql
```

命令提示符变成 *mysql>*，表示你在 *mysql* 程序中直到使用 *quit* 命令退出。

要更新授权表格，输入：

```
UPDATE user SET Password=PASSWORD("your_password"    ) WHERE User= "root";
```

通过 *mysql* 程序发出这条命令时，就是发出了一条 SQL 命令，更新数据库 *mysql* 中的表格 *user*。重置根口令后，退出 *mysql* 并告诉 MySQL 用下面的命令重新读入授权表格：

```
mysqladmin -u root reload
```

现在只要使用正确的根口令，就可以重新进入 *mysql* 程序或其他客户程序。要在 *mysql* 数据库上重新启动 *mysql* 程序，命令是：

```
mysql --user=root --password mysql
```

这将提示你输入口令。如果你在命令行输入口令，而不是让 *mysql* 提示你输入，系统上的其他用户（或黑客）就可以看到你的口令。

如果在 Linux 系统上从 RPM 安装 MySQL，安装期间，*mysql* 用户 ID会被加入系统中 。

这个用户应拥有 MySQL数据目录及其子目录。MySQL守护进程作为一个由系统用户 *mysql* 启动的进程运行，而且对数据库的访问受那个用户的控制。可以用 Unix 的 *passwd* 命令把系统口令设置成用户 *mysql* 的根口令。要对这个用户设置 MySQL 口令，需要使用 SQL命令将用户 *mysql* 插入授权表格中。创建 *mysql* 用户并向其授权访问你的所有数据库的 SQL 语句是：

```
INSERT INTO user VALUES  ( "localhost","mysql",PASSWORD( "your_password"),
"Y","Y","Y","Y","Y","Y","Y","Y","Y","Y","Y","Y","Y","Y");}
```

想对 MySQL 数据库的管理和安全性有更多了解，我们建议你查阅文献目录中列出的相关书籍。

建立数据目录

如果从RPM 安装 MySQL，数据目录会自动位于 */var/lib/mysql*。建立自己的工作站时，你或许没有在 */var* 部分留下很多空间。如果将要用数据库进行很多工作，你可能想给 MySQL 数据目录增加一些空间。

一个简单的办法就是把数据目录移到另外一个部分，建立从那个目录到 */var/lib/mysql*的符号链接。如果这样重新定位数据目录，就不必改变任何MySQL配置信息。

首先，为你的数据目录选择一个位置。例如，可以创建一个 */home/mysql/data* 目录。然后，用下面的命令关闭 MySQL 守护进程（注 4）：

```
mysqladmin shutdown
```

使用 Unix 的 *mv* 命令，把 */var/lib/mysql* 中的所有文件移至 */home/mysql/data*。一旦 */var/lib/mysql* 目录被清空，用 *rmdir*将其删除，用 *cd*进入 */home/mysql* 目录并键入：

```
chown -Rf mysql :mysql data
```

这样就设置了目录中全部文件的正确的文件所有权。最后，用 *ln -s*在 */home/mysql/data* 和 */var/lib/mysql* 两个目录之间建立符号链接。然后键入：

注 4： 还需要在 *mysqladmin* 命令行中包含一 *user=mysql --password*，但从现在开始，我们假定你知道这一点。

```
safe_mysqld &
```

重新启动 MySQL 服务器。可能需要登录为超级用户来进行这些工作。

创建新的数据库

一旦安装并运行了 MySQL服务器，你需要创建新的数据库，并给予自己恰当的权限来读、写数据库。可以作为 MySQL 用户 *mysql*进行这些工作，除非你想立即为自己创建一个独立的身份。我们将在 Web上建立一个生物信息学资源的数据库，因此需要创建一个称为 *resourcedb* 的数据库。要完成这个工作，只需要键入：

```
mysqladmin --user=mysql --password create resourcedb
```

然后用下面的命令在 *resourcedb* 数据库运行 *mysql*:

```
mysql --user=mysql --password resourcedb
```

数据库设计

我们将展示的是一个简单的例子，介绍如何使用 MySQL建立一个生物信息学软件网站的可搜索的数据库（注 5）。我们将讨论一种称为 PHP的脚本语言，PHP允许嵌套命令，使其他人可以直接进入 HTML文件来访问数据库。我们还将讨论从 Web 访问数据库的其他方法。

如果正在从 Web上寻找生物信息学或计算生物学软件，你可能想知道一些关于你所找到的每个条目的信息，还有一些查询数据库的方法。你会想知道每个条目的名称、获得其功能的说明以及下载它的 URL。你可能想知道某个条目的作者和他发表的有关论文。你甚至想可以立刻使用 Medline 链接得到每一篇参考文献。你想知道每个条目是在什么操作系统下工作的，它的有效格式。你可能甚至想有一个到档案文件的直接链接。你可能还想能用“sequence alignment”或“electrostatic potential”这样的关键字来搜索数据库。

注 5： 不要仅仅因为我们在讨论如何建立的问题而在你的计算机上运行和实现这个数据库。Web上充斥着过时的生物信息学链接（和其他类型的链接），除非你是管理员，否则，没人需要你再去加了。

听起来相当简单，对吗？你可能在想所有那些信息最好能进入一个表格，这样就不必在这种数据库中使用复杂的 RDBMS。图 13-3 所示的就是这样的一个大表。

Package Name	URL	Descrip	Keyword	OS	Format	Reference
package name	url	descrip	keyword	os	format	reference
package name	url	descrip	keyword	os	format	reference
package name	url	descrip	keyword	os	format	reference
package name	url	descrip	keyword	os	format	reference
package name	url	descrip	keyword	os	format	reference
package name	url	descrip	keyword	os	format	reference
package name	url	descrip	keyword	os	format	reference

图13-3　如同一个大表的生物信息学软件数据库

然而，如果看得更仔细些，你会发现即使是一个简单的生物信息学软件数据库也不可能真的放在一个表中。记住，表中的数据必须是最基本的，即每个单元格必须只包含一个单独的条目，而不是一个列表或集合。

如果仔细考虑其可能性，你会认识到在生物信息学软件数据库记录中有几处可能出现列表或集合：可能有多个作者或描述软件的多个出版物；软件可能可以用在许多不同的操作系统中；描述每个条目的可能有一个以上的关键字。

关于实体和属性

数据库可包含两种信息：表示一个实体（entity）或可能与其他事物有联系的事物的信息；以及对单个实体的纯粹描述信息，即那个实体的属性（attribute）。

在我们的数据库例子中，我们确定的一点就是一个软件包就是一个实体。让我们开始通过列出与每一个软件包相关的所有信息，来设计这个数据库中的表：

```
Software package name
Software URL
Textual description of function
Operating system
Software format
Archive filename
Reference
Author
Medline link
```

我们可以考虑关于数据库中每一个软件包的更多信息，但对于这个例子来说，这样就足够了。

实体可以用属性和与其他实体的关系来描述。如果数据库中的一个实体没有属性也没有关系，我们就不认为它是实体。在每个软件包的信息列表中的条目一定具有属性；每个参考文献都有一个或几个作者，每个参考文献都有一个 Medline 链接。因此，参考文献应是数据库中的一个独立实体。所以我们需要至少两个表：

```
SoftwarePackage
----------------
Software package ID
Software package name
Software URL
Textual description of function
Descriptive keyword
Operating system
Software format
Archive filename

Reference
---------
Reference ID
Reference name
Reference year
Author
Medline link
```

我们在每个表格中都包含一个“identifier（标识符）”属性。为什么？设想有两个名叫“BLAST”的软件包，它们所做的工作完全不同。在我们的数据库中需把它们区分开，但并不是通过创建另一个“things named BLAST”的表。惟一的 ID 使我们能在同一个表格中存储关于两个名叫 BLAST 的软件包的信息，并能把它们区分开。

理想情况下，我们希望实体之间具有一对一或一对多的关系。参考文献对软件包的关系是一对多的：每个软件包可被多个参考文献描述，但每个参考文献只描述一个软件包（见图 13-4）。多对多的关系无法用 RDBMS 软件处理，因此在建立数据库之前要把它们从数据模型中删去。

如果你的观察力足够敏锐，你会注意到在 Reference 表中，可能会出现多对多的关系。每个作者可以出版多个参考文献，而且每个参考文献可以有多个作者。这种多对多关系的出现表明 Author 可能是一个独立实体，即使我们没有选择在我们现在的数据模型中存储关于作者的任何属性。因此我们实际上需要第三个表：

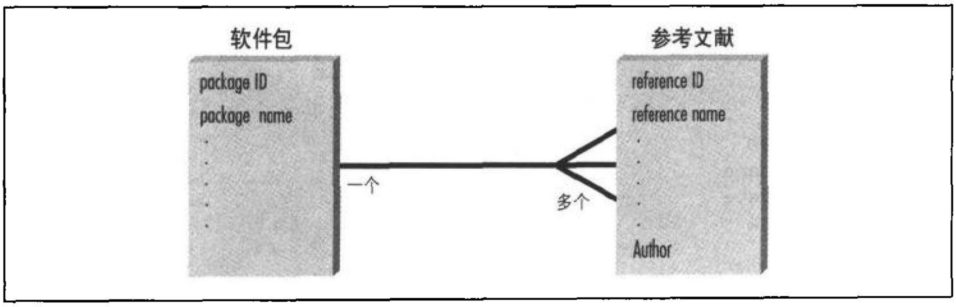


图 13-4　软件包与参考文献的关系

```
Reference
---------
Reference ID
Reference name
Medline link

Author
---------
Author ID
Author Name
```

即使在我们为Author实体创建了一个新表之后，作者和参考文献之间的关系仍是多对多的。这可以通过创建一个连接实体来解决，这个连接实体只是为了解决两者间的多对多关系。连接实体可被称为AuthorRef或其他任意名称。其惟一的属性就是它的惟一的标识符（主键）和建立与其他表的关系的外键。

从数据模型创建数据库

实际创建数据库时，实体就变成了表。每个属性变成表中的一列，ID成为表的主键。与其他表中信息的关系成为外键。

建立关系之前，数据库中的四个表包含下面的信息:

```
SoftwarePackage
----------------
Software package ID
Software package name
Software URL
Textual description of function
Descriptive keyword
Operating system
```

```
Software format
Archive filename

Reference
----------------
Reference ID
Reference name
Reference date
Medline link

AuthorRef
---------
AuthorRef ID

Author
---------
Author ID
Author Name
```

每个属性就是表中的一列，每一列必定具有一种数据类型。主键可以是整数，但不能是 NULL 或空。主键标识符正确的数据类型就是 INT_NOT_NULL；其余字段可定义为某个长度的 TEXT 字段。

创建表间的关系

为存储数据库表间的关系，要把一个表的主键放进另一个表的一列中，那一列称为外键（foreign key）。在我们的例子中，SoftwarePackage 表的主键也被输入到 Reference表的一列中，因为一个软件包对应着多个参考文献。Reference 表的主键和 Author 表的主键成为 AuthorRef 表的外键，每个作者有多个 AuthorRef，每个参考文献也有多个 AuthorRef。

一旦设计出表中将包含什么信息，以及每一列将采用什么格式，你就有了一个所谓的数据库物理模型，就可以像我们前面示范的那样，用 SQL CREATE语句创建数据库了。

开发与数据库互动的基于 Web 的软件

公共生物学数据库的目的是使生物学界能直接共享数据。没有什么比Web更直接的了。因此，开发数据库（尤其是联邦基金资助的）时，最终将考虑怎样使数据能从

Web 上得到，这几乎成为一种约定。有几种技术能实现网页和数据库之间的通信。其中最老的一种称为 CGI（Common Gateway Interface 公共网关接口）编程，不过 CGI 现在正在被 XML和 PHP等其他技术扩充。

基于 Web 的软件开发是一个变化迅速的领域，本书我们不想详细讨论所有的技术。不过，应该知道这些技术是什么，大概了解它们是怎样工作的，因为每次提出请求引导 Web 服务器处理某些信息时就在使用它们中的一种。

如果想建立自己的 Web服务器并向其他用户提供数据分析服务，就需要使用 CGI脚本或混合使用 XML 或 PHP代码的网页。简要介绍 CGI和 XML之后，我们将举一些例子，介绍如何在网页中使用 PHP 命令访问我们在前面刚刚建立的范例数据库。

CGI

CGI 程序，或脚本（script），是一种驻留在 Web服务器上的应用软件。 CGI程序被 Web 服务器的远程用户调用时，就在服务器上执行应用，然后以网页形式将信息传回远程用户，如图 13-5所示。和普通 HTML 网页一样，可以用 HTTP（Hypertext Transport Protocol，超文本传输协议）访问 CGI程序，不过，CGI脚本既可以位于服务器 Web文档目录的专门目录中（如 */cgi* 或 */cgi-bin*），也可以带有 *.cgi* 这样的专门文件扩展名。服务器收到一个 HTTP 请求时，将执行 CGI程序，而不是像在普通网页中那样仅仅向浏览器发送 CGI代码。CGI是一种相对成熟的技术，可被所有主要的 Web服务器支持。

CGI 程序通常由多个部分组成（见图 13-5）。首先，可能有一部分程序从 Web 表单收集用户的输入信息。接下去的一部分程序接受用户的输入信息并进行处理。 CGI 程序可能包含完整的处理输入信息的代码，但更有可能是将输入信息适当格式化并传递给服务器中的一个独立的程序，然后在运行完成后从程序收集输出信息。 CGI 程序最后的功能就是将服务器运行的输出信息以网页形式返回给用户，这个网页可能含有输出的文字信息，也可能是可下载结果文件的链接，或二者兼有。

你可能会用到的一个 CGI程序的例子就是 NCBI 的 BLAST服务器。从 NCBI BLAST 主页选择“ Basic BLAST search ”时，你会注意到新网页的 URL 实际上指向一个 CGI脚本：

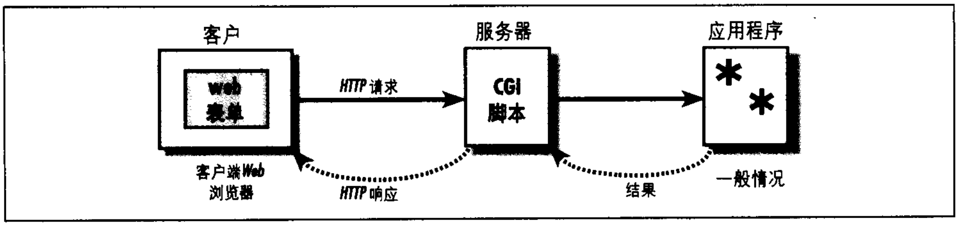


图 13-5　如何执行 CGI 程序

```
http://www.ncbi.nlm.nih.gov/blast/blast.cgi?Jform=0
```

URL 的第一部分，直到问号为止，给出了到 CGI 程序的方向。URL 的第二部分是状态信息（state information），告 诉 CGI 需要它的哪一部分功能。这个 URL 中的状态信息告诉BLAST CGI程序生成一个新的搜索表单，可以向其中输入你的序列。

点 击" Submit "按钮会出现一个新的页面。新页面列出你的请求 ID 以及处理这个请求大约需要多长时间。在后台， CGI 程序将请求传递给在 NCBI 服务器上运行的 BLAST程序。当 BLAST对请求运行结束后，结果输出并以服务器分配的请求 ID作为标识。然后 CGI 程序在那个请求 ID 下查找结果。

搜索运行后，可以选择显示数据。URL仍指向 BLAST CGI 程序，但状态信息改变了。URL 现在如下所示:

```
http://www.ncbi.nlm.nih.gov/blast/blast.cgi?RID=965246273-2980-7926
&DESCRIPTIONS=100&ALIGNMENTS=50&ALIGNMENT_VIEW=0&&HTML=on&OVERVIEW=on
&REFRESH_DELAY=22
```

在这个 URL中传递的状态信息告诉程序哪一个NCBI请求 ID（RID）在服务器上搜索，结果应如何显示，以及在当前表单上通过菜单输入选项的信息。用这个 URL 显示的新页面包含你的 BLAST 结果和到 NCBI 上其他信息的链接的列表。BLAST 结果和链接在NCBI 服务器的后台生成，并写入你所看到的普通网页中（见图 13-6）。

虽然其他生成动态内容的方法越来越流行，但目前Web上大量存在的动态内容仍是用 CGI程序生成的。

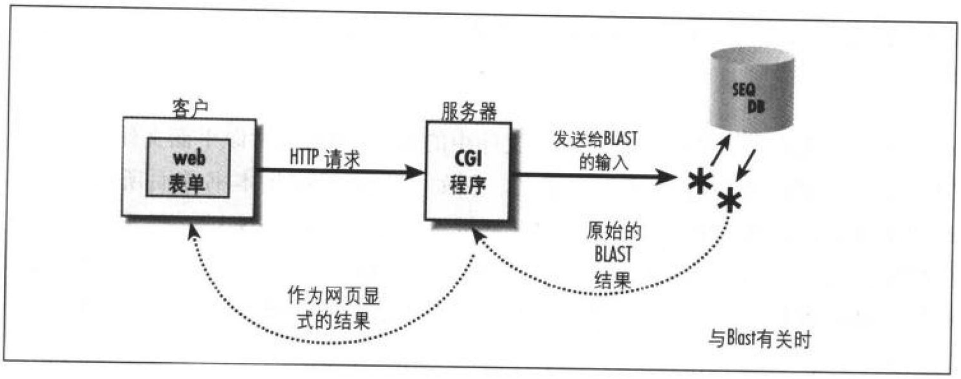


图 13-6　在 NCBI 上处理 BLAST 请求

XML

XML（eXtensible Markup Language，可扩展标记语言）是一种数据显示模式，近年来吸引了人们大量的注意力。与描述网页的HTML语言相似，XML也是从SGML（Standard Generalized Markup Language，标准通用标记语言）派生来的。HTML和XML定义了用来注释文档的标签（tag）。标签由尖括号包围住，语法是*<tags>text</tag>*。HTML标签确定了网页的结构和外观。比如，文本*<B>this is bold</B>*会变成黑体。

另一方面，XML标签定义了文件的内容。例如，在下面的文本中：

```
homologs of the sequence <gi>g7290345</gi>
```

GenBank ID g7290345因为用*<gi>*标签括起来而可以被清楚地识别。如果编写一个搜索文档GenBank ID的程序，如果它们被明显地标记出就会比指定一个GenBank ID正则表达式更容易找到。这样，XML适于采用平面文件数据结构，因而可用标准方式读和写。

给定的XML文档中使用的标签是在DTD（document type definition，文件类型定义）中确定的。DTD起着数据格式词典的作用，指定出现在文件中的元素以及每个属性可能具有的值。DTD可以存在于它自己的文件中，或作为XML数据文件的一部分存在。因为XML允许用户在DTD中定义自己的标签，它为以自由文本（例如，PDB文件中的*REMARK*字段）方式终止的信息提供了丰富而详尽的说明。值得注

意的是，细节过多可使记录迅速膨胀，尤其是表示像微阵列实验的结果这样的复杂数据时。

XML可用标准而一致的方式标记平面文件中的数据,这对通常以平面文件形式存储的生物学数据的处理具有重要的意义。例如，如果想使用本书前面所描述过的ASN.1格式的数据，就需要一个ASN.1解析器，它只读取ASN.1数据。同样，如果想把数据读入PDB格式文件，就需要PDB 文件的解析器。然而，如果数据用XML标记，XML解析器就可以把数据读入程序。下面是一个PDB作者数据的XML表示的例子：

```
< !--简单的 PDB 引用 DTD -->
<!ELEMENT citation (author)+>
<!ELEMENT author (first-name, last-name)>
<!ELEMENT first-name (#PCDATA)>
<!ELEMENT last-name (#PCDATA)>

<!DOCTYPE pdbcite SYSTEM "pdbcite.dtd">
<citation>
   <author>
     <name>Fermi, G.</name>
     <citation_id>primary</citation_id>
   </author>
   <author>
     <name></name>
     <citation_id></citation_id>
   </author>
  ...
</citation>
```

许多XML解析器对生物信息学研究中常用的编程语言，包括Perl、Java、Python和C++，都是适用的。XML解析器有两种基本类型：非验证型(nonvalidating)和验证型(validating)。非验证型解析器读取XML文件并保证其语法全部正确。相反，验证型解析器可以防止遗漏或出现错误数据。把XML文件与其DTD比较，验证型解析器不仅可保证数据标记的语法正确，而且确保每一个标签或属性与恰当的值相关联。

XML应用

得益于灵活性和在其他领域的成功，一些生物信息学项目采用XML作为数据描述语言。特别是XML很好地应用在基因组注释（genome annotation）中：Berkeley

Drosophila Genome Project的Suzanne Lewis及合作者们开发了 Genome Annotation Markup Element（GAME-XML）DTD来表示基因组中的序列特征。 XML 也是 Lincoln Stein、Sean Eddy和 Robin Dowell为分布式注释系统（distributed annotation system, DAS）提出的标记方案的基础。其他一些 XML应用的例子包括在 Proteometrics 开发的 Biopolymer Markup Language （BioML）序列描述格式，Albany 大学的 Ronald Gilmour开发的用于表示分类树的拓扑学的 Taxonomic Markup Language，以及表示小分子结构的 Chemical Markup Language （CML）。

关于XML在生物信息学中的这些及其他应用的信息可以从 Bioperl Project的 XML 分部 Bioxml组的主页上找到（*http://www.bioxml.org*）。XML的其他信息及相关技术可以从万维网联盟（*http://www.w3c.org*）得到。

PHP

PHP 是一种用于 Web 服务器的超文本预处理器模块，它使 Web 服务器能读取和解释嵌在网页中的 PHP代码。PHP代码类似但不等同于常见的编程语言，如 Perl和C。

PHP 在大多数 Web 服务器上运行，更多信息参见 *http://www.php.net*。与其他一些动态内容技术（例如 Microsoft 的 ASP）不同，PHP是一个被多种操作系统支持的源代码开放的方案。PHP还内置支持与 MySQL、PostgreSQL和 Oracle数据库的相互作用。

从Web服务器上请求一个含有 PHP代码的网页时，服务器在网页上处理 PHP指令，然后传递给客户。网页源代码以标准 HTML形式出现在客户端，在 Web服务器之外的计算机上看不到 PHP代码。

PHP命令可以与HTML代码以有利于网页设计者的任意顺序交替排列，结果 HTML 将在 PHP 代码处理之后出现。PHP 还能包含外部文件，因此可以编写代码控制主 PHP文件中网页的外观和独立文件中加入子例程的位置。 PHP页面用 *.php*扩展名区别于标准 HTML文件。

用 PHP 访问 MySQL数据库

用 PHP访问 MySQL 数据库并不复杂。需要一行代码连接数据库，用另一行代码选

择要使用的数据库，一个查询语句，以及一个把数据以 HTML 形式发送给客户端 Web 浏览器的语句。下面是一个简单的例子：

```
<?php
$link =@mysql_pconnect ("myhost.biotech.vt.edu","cgibas","password") or
exit ();
mysql_select_db ("resourcedb") or exit ();
$result = mysql_query ("SELECT program, url, institution FROM software
WHERE program = "BLAST") or exit ();
while ($row = mysql_fetch_row ($result))
{
    print("<br>\n");
    for ($i = 0; $i < mysql_num_fields ($result); $i++)
    {
       print ($row[$i]);
    }
}
mysql_free_result ($result);
?>
```

第一行代码(*<?php*)表示一块 PHP 程序开始了。下一步是用指定的名字和口令连接到 MySQL 服务器上，或如果连接失败就结束程序：

```
$link =@mysql_pconnect ("myhost.biotech.vt.edu","cgibas","password") or
exit ();
```

现在请求称为 *resourcedb* 的数据库：

```
mysql_select_db ("resourcedb") or exit ();
```

下一步进行MySQL 查询，当程序名称为‘BLAST’时，选择*software*表的 program、URL 和 institution 字段的值：

```
$result = mysql_query ("SELECT program, url, institution FROM software
WHERE program = "BLAST") or exit ();
```

每从数据库中读取一行，就把那一行分成若干字段，并把它们赋给 $ *row* 变量，然后在这一行的每个字段单步执行程序，并打印出该字段的值（注 6）：

```
while ($row = mysql_fetch_row ($result))
{
    print("<br>\n");
```

注 6：　用这种方法会打印输出比较难看的无格式文本。在命令序列中的不同位置加入HTML 表格标签就会得到美观得多的输出结果。

```
    for ($i = 0; $i < mysql_num_fields ($result); $i++)
    {
       print ($row[$i]);
    }
}
```

最后，查询结束后从内存中释放结果：

```
mysql_free_result  ($result);
```

程序的最后一行（? >）终止网页的 PHP 部分。

用 PHP 从表单收集信息

显然，如果用一个变量名称代替"BLAST"一词，并制作一个小表单允许用户输入一个词赋给变量名称，这个程序就会更有用。出乎意料的是，这样我们就有了一个能在 *resourcedb* 数据库中寻找与用户有关的程序的粗糙的搜索引擎，而不是在数据库中搜索 BLAST 服务器的 PHP 小程序。

用 PHP *print* 语句生成表单的 HTML 代码来创建表单。例如，生成一对单选按钮的 PHP 代码如下：

```
print("<INPUT TYPE=\"radio\" NAME=\"type\" VALUE=\"Yes\" CHECKED>Yes\n");
print("<INPUT TYPE=\"radio\" NAME=\"type\" VALUE=\"No\">No\n");
```

其他表单特征用类似的方法实现。关于表单的更多信息，用表单收集数据以及如何生成 PHP 表单的详细例子，见参考文献中的 MySQL 参考文献。

Web 数据库编程并不是从几页书中就能学会的，但我们希望我们已使你相信，如果需要，建立一个 MySQL 数据库是你能做到的事，而且编写 PHP 程序来访问这个数据库并不比使用 HTML 和 Perl 更难。我们没有展示 MySQL 数据库例子的完整 PHP 程序，而只是选取了其中比较重要的部分。

与范例数据库相互作用的 PHP 程序主要应完成以下几项工作：

1. 向用户显示一个欢迎页面。这个页面应允许用户选择搜索数据库或是向数据库中添加新的条目。在后台，PHP 程序对那个选择进行处理，以便随后显示正确的页面。

2. **向用户显示一个查询提交表单。**PHP程序需要生成一个有效的表单，然后获取用户输入表单的数据并用它生成一个 SQL SELECT语句。

3. **向用户显示查询结果。**在数据库中找到匹配记录时，程序必须把每个记录格式化为相当美观的 HTML代码，以便在用户的 Web浏览器上清楚地显示。

4. **显示添加新条目的表单。**假定你有向面向外部用户的数据库增加条目的权限，而这要求你提交用户名和口令信息。

5. **向数据库添加条目。**这个例程需要从追加表单中获取信息，并使用 SQL INSERT命令将其加入数据库。

第十四章

可视化和数据采集

生物信息学中的任何结果，无论是序列比对、结构预测，还是基因表达模式的分析，都应解决一个生物学问题。因此，研究者要负责针对明确的问题解释他们的结果，并使他们的同行能得到那些结果。这个解释的步骤是科学过程中最重要的部分。要使你的结果有用，它们就必须是可解释的。我们重申一次：如果结果不能被解释，它们就无助于任何人，甚至是你自己。

本章我们将介绍一些能帮助了解结果的意义的计算工具。为此，我们大致按照对应于数据分析过程的顺序组织本章的内容。本章第一部分介绍一些用于生物信息学研究各种数据的可视化程序。这些程序包括数字数据的通用绘图和数字数据统计软件包，以及专门用于以可解释的形式显示序列和结构信息的程序。本章第二部分介绍一些用于生物信息学数据采集（在大批数据中查找、解释和探寻模式）的工具。

本章涉及的主题基本上是计算统计领域的一些分支。正如前面的章节中所提到的，统计方法为检验研究者的发现的意义提供了一种手段。人的神经系统对图形模式非常敏感。如果仔细观察一个足够长的蛋白质序列，慢慢就会开始发现它的图形模式，它们在氨基酸序列中或许具有生物学意义（像家族标志序列的一部分，如氯霉素乙酰转移酶中的 P.YTVF），或许没有（单词或名称，如 PER，注 1）。因此，我们用可视化来开发眼睛和大脑发现可能有意义图案的能力。用统计和数据采集来尽量减少直觉的影响，并把我们的搜索限制在那些具有定量意义而且可重复的图形上。

注 1：　开始看到单词或人名形式的序列基序时就应该停下来想一想。

准备数据

准备数据[也称预处理（preprocessing）或数据纯化（data cleansing)] 是数据采集中最重要的一部分，也是最没有吸引力、最少讨论的部分。预处理可以只是确定数据是否符合程序所要求的格式，也可以包括大量的计算。

作为生物信息学研究者，尤其必须注意你的数据。你的结果和信誉取决于其他研究者所提供的数据。因此，你必须谨慎地收集和利用那些数据。下面列出了进行设计时要回答的一些有关数据完整性的问题（这个列表并不完整，具体到某个项目可能还有其他问题要考虑):

- 你的数据是否是你所期望的？例如， DNA 序列应该只包含 A、T、C 和 G，除非程序理解其他 IUPAC （国际理论化学和应用化学联合会）符号。蛋白质序列应只包含 20种氨基酸。可以用 *grep* 快速检查文件中是否包含有错误字符的行。
- 你的数据文件是否符合所要使用的程序要求的格式？谨防机动性更强的格式。例如，有些程序对 FASTA 文件中的注解行有长度限制，而有些没有。
- 注意序列的多种变体。剪接变体、突变、删除、测序错误、不小心截断序列文本都可能导致得不到你想要的序列。这就需要注意哪些序列或结构的变化是与生物学相关的，哪些变化是由实验过程中的人为因素引起的。
- 确定能从 GenBank 或其他数据库找到要处理的序列，除非向你提供这些序列的合作者还未把它们保存到数据库中。
- 处理大型表格式数据时，确定表格中的每个字段都具有适当的值。使用 XGobi 这样的程序是比较好的检验方法，因为如果不是每个字段都具有一个值，程序就会发出提示。如果值超出了范围，结果就会成为界外值，XGobi 等可视化工具也可以对其进行检验。
- 程序对测试数据是否产生有意义的结果？应用新程序时，应该用一些已知结果的数据来测试程序，确定它给出了正确的结果以及以合理的方式运行（这种测试有时称作健全检查，*sanity check*）。比如，如果一个程序对两个序列或结构进行比较，无论按何种顺序输入数据能否得到相同的结果？
- 检验所使用程序的副作用。程序是否更改其输入数据？这种更改可能比较隐蔽，例如在序列文件中每 10 个残基之间加入空格，将数值标准化，或改变结构数据的坐标值。

对于微阵列数据，其值是否已标准化？如果你使用的程序完成任何类型的标准化，确定如何实现标准化是很重要的。

对于蛋白质结构数据，所有原子编号和残基编号是否连续？结构完整还是含有链中断或其他异常？所有残基是否都以标准氨基酸三字母代码标记？所有原子是否都以正确的 PDB 原子代码标记？

最后，确定理解所有用到的程序，至少能知道它们的输入、输出数据的格式和意义。

浏览图形

如果你打算处理图像，就需要一些 Linux 和 Unix 系统下的基本的浏览图形文件的工具。有许多图形程序可供使用，下面我们要介绍的三种是最为常见、使用简单而且免费的。

xzgv

xzgv 是一种在 X Window 系统中查看图形文件的程序。它可以显示更流行的图形格式（GIF、PNG、JPEG），以及其他多种格式。作为简单的图形查看程序，它具有一些灵活的功能。例如，创建缩略图（预览文件的小型图片）和像放映幻灯片一样逐个显示图像。

Ghostview 和 gv

Ghostview 和 *gv* 是 PostScript 和 PDF（Portable Document Format）文件的查看程序。PostScript 和 PDF 是 Adobe 公司开发的页面描述语言（注 2）。两个程序都允许翻阅文档、跳至某一特定页面、打印整个文档或选定的页面，以及其他简单的文档导航任务。越来越多的期刊以 PDF 文件的形式发行电子版论文，因此 gv 这样的文档查看器就非常有用了。

注 2： Adobe 还开发了一种 PDF 阅读程序，称为 Acrobat Reader，可以在 Windows、Mac OS、Linux 和少数 Unix 系统上免费使用。

由于能够生成压缩程度更高的文件并与Windows应用软件相配合，PDF似乎正在超过PostScript成为更常用的格式。网络上许多文件仍是以PostScript格式传递的，包括未定稿（尤其是那些使用 LaTEX 文档格式语言的作者所写的未定稿）以及一些基于 Web 的服务器的输出结果（例如 BMERC 二级结构预测服务器）。

GIMP

GIMP（GUN Image Manipulation Program 的缩写）是一种图像处理软件包，与 Adobe Photoshop 具有类似的功能和特征。尽管 GIMP 能打开并查看图形文件，但这样似乎有些大材小用。然而，改换格式或修改图形以用于演示文稿或论文时，手头有图像处理软件将会是雪中送炭。

序列数据可视化

在第八章中我们介绍过查看序列数据（特别是多序列比对）的工具。正如我们在那一章中提到的，从序列比对中迅速汇总信息的最好方法之一就是使用序列标志。本节我们将讨论一种序列关系可视化的几何学方法，并介绍 TEXshade，一种生成可达到出版物质量水平的序列比对图形的程序。

用 TEXshade 进行出版物质量水平的序列比对

TEXshade（*http://homepages.uni-tuebingen.de/beitz/tse.html*）是一种用 LaTEX 这个为数学家和计算机科学家发明的文档标记语言编写的用于标出序列比对的软件包。这个软件包非常灵活，你可以根据保守性和化学特性对比对的残基进行填色和做标记。而且，TEXshade 能够连接二级结构和从 DSSP 程序输出得到的信息（第六章介绍过），以及由 PHD预测服务器产生的预测二级结构的结果。最后，TEXshade 能自动产生对一个序列比对提供提纲挈领式查看的“指纹”，在指纹中，保守残基所在的列用垂直行表示。与序列标志相似，指纹能迅速概括比对数据，发现毗邻的保守残基群。

TEXshade的名称来自 LaTEX 文档。如果把序列比对以 MSF格式（GCG多序列比对格式）存储在文件 *alignment.msf*中，下面的 LaTEX 文档将生成一个 TEXshade 默认格式的比对：

```
\documentclass{report}
\usepackage{texshade}
\begin{document}

\begin{texshade}{alignment.msf}
\end{texshade}

\end{document}
```

LaTeX 是一种类似于 HTML 的文档标记语言。在上述例子中（其输出结果如图 14-1 所示），我们告诉 TeX 翻译器这个文档有开头和结尾，而且在 *alignment.msf* 中只含有一个 TeXshade 序列比对。我们需要对最后得到的比对进行手工标记。如果这个比较费力，SDSC 生物学工作台提供了一个到 TeXshade 的图形界面，可以把 TeXshade 格式的序列比对自动转换成 GIF 或 PostScript 图像。

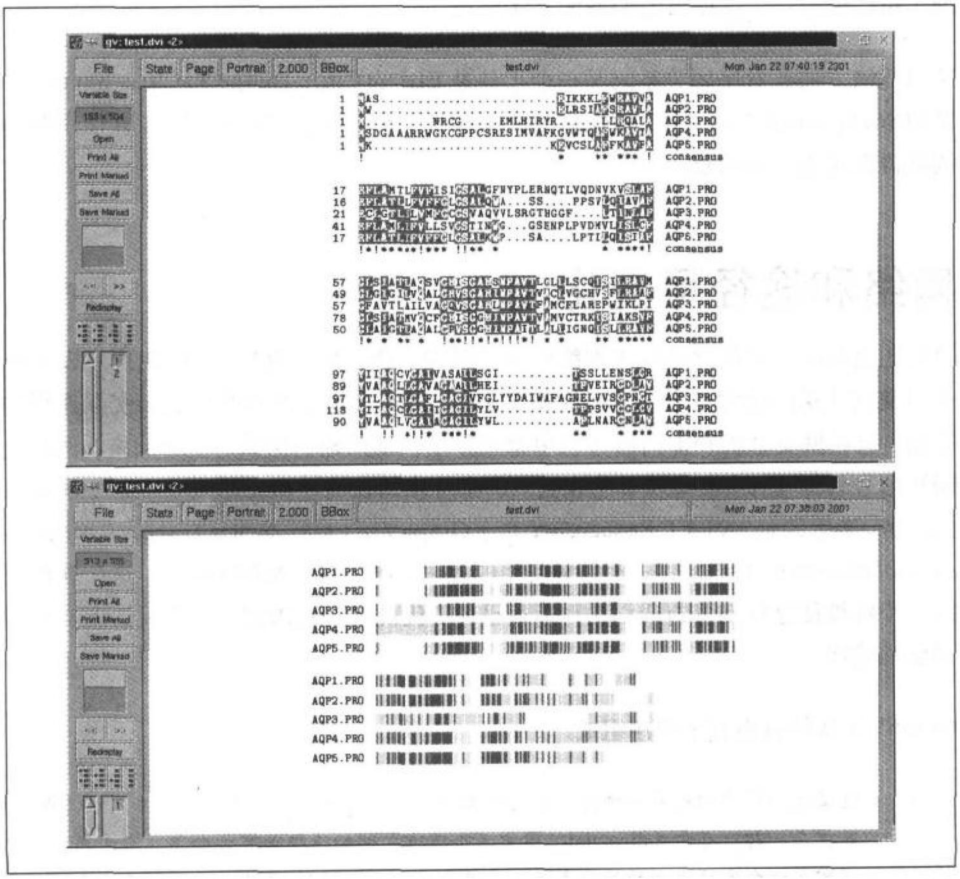


图 14-1　TeXshade 比对及相应的指纹

用几何学方法估计序列距离

多序列比对和序列标志代表一组比对序列在每个位置上的相似性。然而即使把保守残基着色，也不总能很容易地辨别序列是如何关联的。有时，用更抽象的方法来看序列如何成组则更有帮助。系统发育树是将关联性可视化的一个方法。

DGEOM是由 Mark J. Forster及合作者开发的一组程序，它能处理一组比对序列（GCG的 MSF格式的多序列比对或一组成对比对），并在三维空间中将其表示为点，点之间的距离代表序列间的进化距离。这些点被写入一个PDB文件，而且可以用你选择的蛋白质结构浏览程序浏览。把序列比对的表示以用于结构数据的文件格式存储的想法可能会使一些人望而却步，但这个程序完成得很好，而且因为高质量结构可视化软件包很容易找到，这种方法省却了对独立图形系统的需要。这些程序是用Perl和 C编写的，这使得它们非常易于修改。

采用几何方法查看序列关系的另一个工具是 EBI的 Chris Dodge和 Chris Sander开发的 SeqSpace软件包。这个软件包包含一些计算序列距离的 C++程序，并使用 Java浏览器表示为三维的点。

网络和途径可视化

目前还没有用于可视化途径或网络中分子间交互作用的标准软件包。用图示表示分子间交互作用的最常见方法是绘制图（注 3）。图还可用于图解其他表示交互作用的数据，包括聚类分析和蛋白质结构中相互作用残基的输出结果。生物网络，例如代谢和信号转导途径，其连接非常密集，创建可读而且高度相关的图并不是一项简单的任务。幸运的是，AT&T Research发布了 GraphViz（*http://www.research.att.com/sw/tools/graphviz/*），这是一套免费使用的程序，可用来绘制和编辑图。这个软件包的三个特性使之特别具有吸引力：运行迅速、优良的文档编制，使用灵活而直观的语法描述图。

GraphViz软件包由五个程序组成：

注 3：这里我们讨论的是某一种图。在这种图中，一系列点（或节点）由一系列线（或边）连接在一起。有向（directed）图中两个节点之间的连接是有方向点。有向图中的边通常用指向连接方向的箭头表示。在无向（undirected）图中，边是没有方向的。

dot

绘制有向图

neato

绘制无向图

dotty

编辑图的图形界面

lefty

编辑图和其他图表的语言；用于编写 *dotty*

tcldot

用 Tcl语言编写的到 *dot*的图形界面

要绘制一小组交互作用的有向图，可以把下列代码输入文本编辑器，并保存为名叫 *morphopath.dot* 的文件：

```
digraph G {
size="3, 3";
    SHH -> "Early gene expression";
    "FGF-4" -> "Early gene expression";
    SHH -> "BMP-2";
    "BMP-2" -> "FGF-4";
    "FGF-4" -> SHH;
    "BMP-2" -> "Late gene expression";
    "FGF-4" -> "Late gene expression";
```

用下面的命令在 shell 提示符下调用 *dot* 程序：

```
% dot -Tps morphopath.dot -o morphopath.ps \end
```

这个命令告诉 GraphViz程序组中的 *dot* 程序生成 *morphopath.dot*所描述图的 PostScript 图像，并把这个图像保存到 *morphopath.ps* 文件中（见图 14-2）。

如果在 C编程语言方面有些经验，你可能知道图描述格式与 C语言中的 struck类似。使用这种描述图设计的结构化方法，就能在图内定义图，而且还能从 Perl脚本直接生成GraphViz 文件。除了特定的连接性外，GraphViz程序生成的图可以用标签、不同类型的箭头和突出的框作注解。

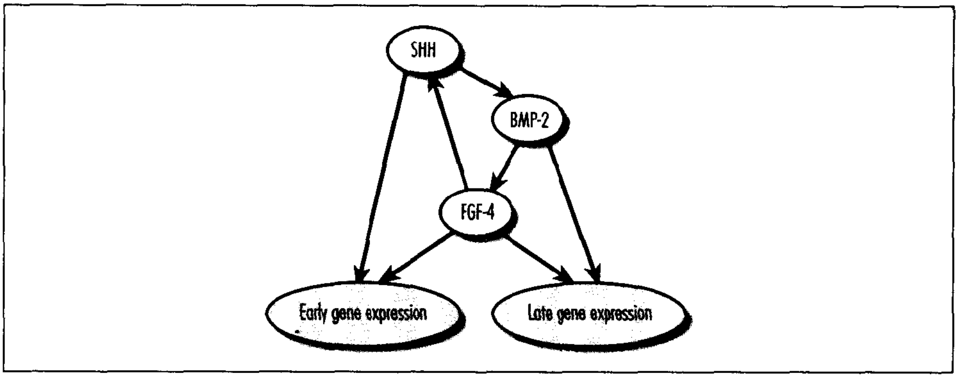


图 14-2 morphopath 途径输出结果

处理数字数据

数字数据总能填入电子制表软件，如 Microsoft Excel 或 StarOffice Calc，并用程序内置的绘图功能绘制图表。通常这是快速绘图的最好方法。不过有时候我们需要对绘图过程进行更多的控制，这就超出了电子制表软件所提供的范围。这种情况有两个常见的例子，一个是修改图表格式用于发表，另一个是处理多维数据集。如果必须用数据生成图表，我们建议你看一看 Edward Tufte所著的关于可视化理解的文献（见参考文献）。其中有许多例子，而且介绍了制作清楚表达意思的图表的技巧。

在这一部分，我们将介绍一些制作图表（plot）的程序。另外，我们还要介绍两个具有良好的可视化和数据分析功能的专门编程语言：统计学语言 R（及类似的商业编程语言 S-plus）和数字编程语言 Matlab（及相应的免费的 Octave）。

gnuplot和 xgfe

gnuplot（*http://www.gnuplot.org*）是在制作科学数据的图表方面更广泛使用的程序之一。由于它具有灵活性、耐久性和开放源代码的特性，*gnuplot*加入了一些功能部件，包括用于在文档中加入图表的脚本和实用程序。不过，有利就有弊，灵活性和耐久性的反面是令人生畏的命令语法。所幸还有一个称为 *xgfe*的到 *gnuplot* 的图形界面。*xgfe*有助于数据或函数的快速绘图，如图 14-3所示。从*http://home.flash.net/~dmishee/xgfe/xgfe.html*处可以找到更多有关 *xgfe* 的信息。

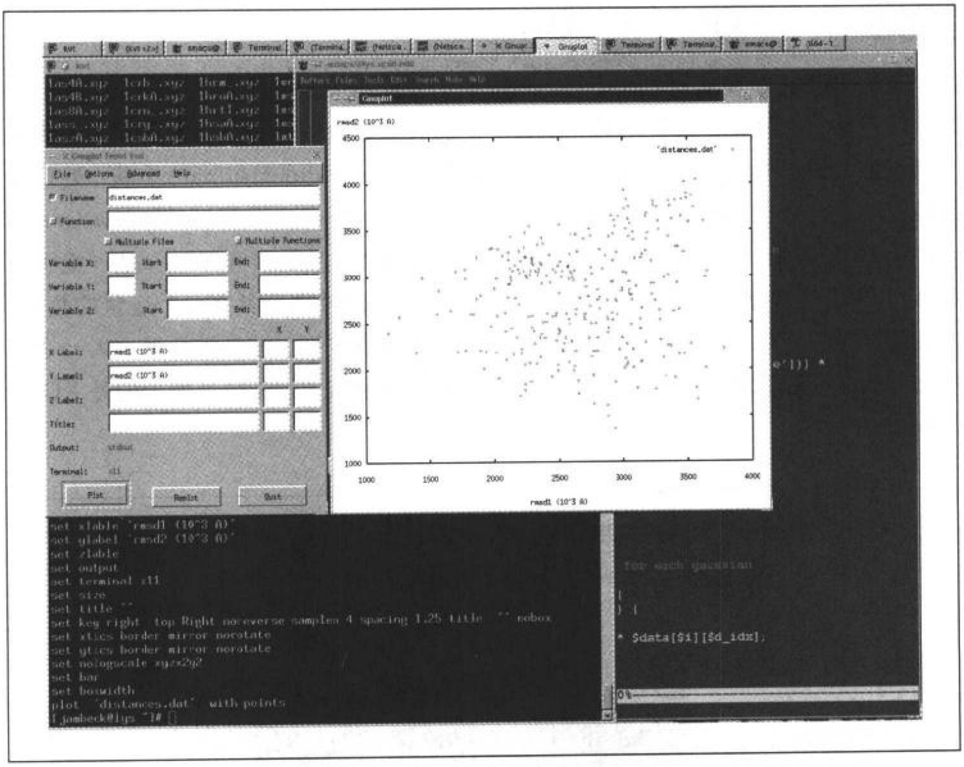


图 14-3　xgfe/gnuplot　输出结果

如果需要对输出结果的格式进行更多的控制，你必须通读*gnuplot*的文档资料，了解它能做什么。另外，如果需要从数据自动生成一些图表，你可能得弄清如何使用 Perl 或其他脚本语言控制 *gnuplot* 的行为。

Grace：数据可视化小工具

Grace（*http://plasma-gate.weizmann.ac.il/Grace/*）及其前身 *xmgr* 作为一种功能强大的二维数据绘图工具，是替代 *gnuplot* 的一个软件。Grace 在 X Window 系统下使用简单的图形界面，可以进行大量的菜单驱动的图表定制。和 *xgfe* 一样，Grace 提供了在 80% 的情况下所需的相当简单的 20% 的功能。除了目前在以色列的 Weizmann Institute of Science 的主要站点发布外（这里是最新版本），还有许多可以得到 Grace 的镜像站点。主站点还有关于使用 Grace 的常见问题和应用指南。

多维分析：XGobi 和 XGvis

如果数据有两三个变量，可以把它们分配到图表的不同坐标轴上，Grace 和 *gnuplot* 等作图程序是适用的。遗憾的是，生物学中的大多数数据具有高得多的维数。研究多维数据的科学称为多元（multivariate）或多维(multidimensional)分析。处理多维数据中的一个重要问题是可视化。XGobi（*http://www.research.att.com/areas/stat/xgobi/*）是为那些无法想像 18 维空间的人设计的。XGobi 和 XGvis 是一组可以从 AT&T 免费获得的程序。XGobi 允许以三维方式同时浏览具有多个变量的数据，它们形成一组可以用鼠标旋转的点。XGvis 完成多维定标，把多维数据挤入一个可视化的空间（通常是一个二维图表或可旋转的三维图表）。图 14-4 所示的是 XGobi 的输出结果。

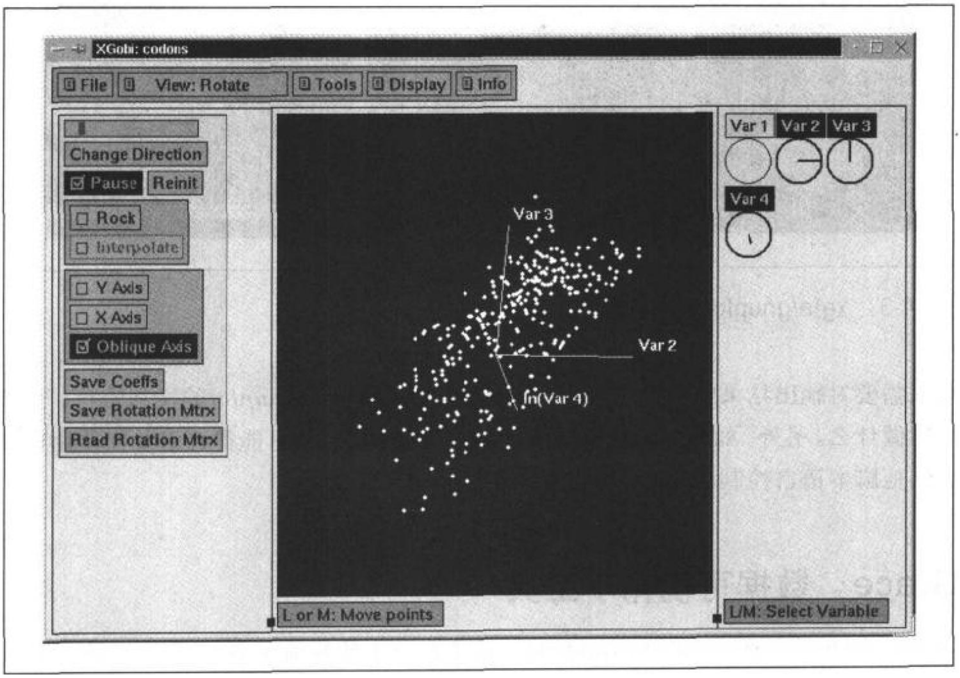


图 14-4　XGobi 实例

XGobi 具有很多功能，这里只是概要的介绍。XGobi 把包含数据列的文本文件作为输入数据。如果你有一个数据文件 *xgdemo.dat*，可以在 shell 提示符下输入下面的命令浏览此文件：

```
% xgobi xgdemo.dat &
```

XGobi最初把这些点以二维散点图的形式呈现出来。从窗口顶部的 View 菜单选择 Rotation显示这些点的活动三维图表，可以通过在这些数据点范围内单击或移动鼠标来控制。从 View 菜单选择 Grand Tour 或 Correlation Tour 可使这些点在数据空间内随机旋转。

变量窗口小部件（variable widget，沿着 XGobi界面右侧的圆）代表数据中的每一个变量。每个小部件中的线条代表图表中变量的轴的方位。如果数据包含三个以上的变量，可以用下面的方法选择要显示的变量：首先单击要去除的变量窗口部件，然后单击要显示的变量窗口部件；最后，单击相应变量的名称，进行一系列的变换（例如自然对数、常用对数、平方、平方根）。和 GraphViz绘图程序一样，XGobi和 XGvis得到大量文献的支持，而且如果参照 XGobi主页上的说明，可以很容易地在 Linux系统上安装。一些 Linux 产品（如 SuSE）甚至包含 XGobi。

数据分析编程

本节我们将介绍两种新的非常适用于数据分析的编程语言。刚刚学过 Perl，学习更多语言的建议似乎有些反常。谁想仅仅为了数据分析而学习整个语言？如果电子数据表和计算器能满足统计的要求，你是用不着这些软件的。而且正如在上一章我们看到的，有一些实用程序可用于用 Perl，特别是 PDL（Perl Data Language，Perl 数据语言）、创建用途广泛的应用程序。

然而，生物信息学中的许多问题需要使用数字或统计计算。开发、调试这样的软件所花费的时间相当可观，而且如果只使用一两次，也不值得为编写这样的程序浪费太多时间。幸运的是，与用 Perl语言可以容易地开发处理数据的程序一样，数据分析语言（没有一个更好的名词）也使得原型设计和数据分析程序的快速开发变得容易起来。下节，我们将介绍一种用于统计的语言 R（及商业版本 S-plus）和一种用于计算数学的语言 Matlab（及免费版本 Octave）。

R和 S-plus

R是在 AT&T Bell 实验室开发的 S 统计编程语言的一个免费工具。R是由 Ross Ihaka 和 Robert Gentleman 在 Auckland 大学开发的。R 及其商业版本（S-plus 3.x、4.x

和 2000)可在 Unix 和 Win32 平台上使用，而且语法相似，因此我们所说的 R 指这两种语言。

R通常在已解释的环境(interpreted environment)中运行。R 自身提供了交互环境，可以逐句运行各语句或把它们作为整个程序运行，而不是编写从命令行执行的整个程序。要启动 R 环境，在命令提示符下输入 R 并回车。你会看到以下几行文字:

```
R : Copyright 2000, The R Development Core Team
Version 1.1.1  (August 15, 2000)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type    "?license" or "?licence" for distribution details.

R is a collaborative project with many contributors.
Type    "?contributors" for a list.

Type    "demo()" for some demos, "help()" for on-line help, or
        "help.start()" for a HTML browser interface to help.
Type    "q()" to quit R.
```

屏幕底部的尖括号（ >)就是 R 提示符，类似于 Unix 中的 shell 提示符。在下面的例子中，我们在用户（你）应该输入的语句之前写入 R 提示符。

首先应该知道两个命令。在任何交互环境中最重要的命令可能就是退出操作系统的命令。在 R 中，退出的命令是:

```
> q()
```

第二个最有用的是提供 R 的在线帮助系统访问的命令 *help()* 。无参数的默认 *help()* 命令，返回到如何使用 *help()* 的说明。如果需要了解关于某一具体 R 函数的帮助，可在 *help* 后面的小括号中加入这个函数的名称。例如，如果想了解 *source()*函数，可以输入:

```
> help(source)
```

还可以使用？作为 *help()* 的简写，在这个例子中，可以只输入？ *source*。在上文我们提到，R 环境是交互式的。如果输入:

```
> 2 + 2
```

R 会告诉你 2+2等于 4:

```
> 2 + 2
[1] 4
```

R 的响应(*4*) 前面是加中括号的数字（ *[1]*)，表明输出矢量中答案的位置。与 Perl 不同，R 没有标量变量。像前面的答案那样简单的数字是存储在一个表示长度的矢量中的。请注意矢量中第一个元素是从 1 而不是 0 开始编号的（注 4)。

<- 运算符是赋值运算符。其功能与 Perl 中的 = 运算符类似。输入 :

```
> a <- 2 + 2
```

不生成输出结果，因为计算的结果存储在赋值运算符左边的变量中。要看一个变量所包含的值，在 R 提示符下输入变量名称:

```
> a <- 2 + 2
> a
[1] 4
```

正如 Perl具有可用于处理文本的基本数据类型一样 , R具有可用于统计的数据类型。我们已经看到了矢量； R 中还有矩阵、数组（矩阵的多维化结果）和因子（标记相同长度矢量的字符串的列表）。

R 的在线资源

从 CRAN（Comprehensive R Archive Network ）上可以获得更多有关 R 的信息。可以通过搜索引擎或到 R Project 主页(*http://www.R-project.org*) 的链接找到离你最近的 CRAN站点。CRAN 中有大量用于在 R 中实现特殊统计应用程序的包，以 RPM 文件的形式提供（有关 RPM 安装的细节见第三章）。如果你的项目要求采样、聚类、回归或因子分析，R能为你解燃眉之急。R还可以作为输出系统直接访问 XGobi，以便计算结果可以绘制为二维或多维图表。

可以尝试不安装而使用 R ,这得益于 Montana州立大学数学系在 Rweb (*http://www.math. montana.edu/Rweb/*) 提供的服务。Rweb 接受你的 R 代码，运行并返回计算结果的页面。如果想用 R进行其他工作而不是简单的示范，那么把 RPM 文件下载到本地计算机并运行 R 会更快些。

注 4: 实际上 , R 矢量是有 0 元素的，但它并不接受赋值 , 而且返回 *numeric(0)* ，这是一个空矢量 。

如果你发现 R 很有用，我们向你强烈推荐 R 的学习指南 An Introduction to R 和 R FAQ（*http://cran.r-project.org/*）以及《Modern Applied Statistics with S-Plus》第三版（见参考文献）。Venables 和 Ripley 现在都是 R 开发核心程序员组的成员，尽管他们的著作主要是关于 S-plus 的，但从 Ripley 的网站（*http://www.stats.ox.ac.uk/~ripley/*）可以得到一些补充内容，使书中的例子能更容易地在 R 下使用。

Matlab 和 Octave

GNU Octave（*http://www.gnu.org/software/octave/octave.html*）是一种免费的编程语言，它的语法和函数都类似于 Matlab 这种 MathWorks 公司（*http:// www mathworks.com/ products/matlab/*）的商业编程环境。Matlab 在快速编写用于大量数字计算的程序的工程师中十分流行。如果要快速编写进行大量复杂计算的程序的原型，特别是那些用于数据分析或模拟的程序，Octave（或 Matlab）十分值得关注。Octave 和 Matlab 都能在 Unix 和 Windows 系统下使用。Octave 源代码、二进制码和文档都可以在线获得；它们还作为不断增加的 Linux 产品的一部分而发布。

Octave 用前面提到的 gnuplot 软件包生成图形输出结果。尽管这个方案运行得很好，但与 Matlab 的绘图能力相比，显得有些简单。实际上，如果你是学生，我们非常鼓励你利用 Matlab，它可以使你事半功倍。下节将要讨论的许多数据采集和机器学习工具都是 Matlab 软件包。

可视化：小结

本章介绍了生物信息学中经常出现的数据显示问题的解决方案。要得到生物信息学在可视化方面的一些最新进展的信息，可参见欧洲生物信息学研究所的网页（*http://industry.ebi.ac.uk*）上的链接提供的可视化信息。本书的网页上有更多可视化和数据采集的在线资源的链接。表 14-1 列出了用于数据可视化的工具和技术。

表 14-1　数据可视化工具和技术

目的	原因	可用工具
浏览图形文件	浏览结果和查看图表	*xzgv*
浏览 PDF 或 PostScript 文件	阅读电子形式的文章	*gv*、Adobe Acrobat Reader

表 14-1　数据可视化工具和技术（续）

目的	原因	可用工具
操作图形文件	制作图形	GIMP
二维或三维数据绘图	汇总显示数据	Grace、*gnuplot*
多维可视化	浏览多于三个变量的数据	XGobi
多维缩放	以二维或三维形式浏览多维数据	XGvis
绘制图形结构	绘制网络和途径	GraphViz程序
清晰打印序列比对	修改序列比对的格式以达到出版要求	TEXshade
涉及大量统计的数据分析程序设计	统计数据分析工具的快速原型设计	R（或 S-plus）
涉及大量数字计算的数据分析程序设计	大量使用矩阵的工具的快速原型设计	GNU Octave （或 Matlab）

数据采集和生物学信息

现代生物学最令人兴奋的领域之一就是数据采集方法在生物学数据库中的应用。这些方法可以同样属于机器学习（machine learning）的范畴，这是在人工智能领域使用的表示一大系列根据经验调整其性能的程序的名称。这里我们列出一些生物信息学最近的研究中所出现的技术，并不完整，但愿能提供一个了解这个正在发展的领域的起点。

注意，对数据采集和机器学习的讨论中有一些拟人化的说法，但这没什么不可思议的。虽然说程序“学习”或被“训练”，但它们总是受一组意义明确的指令控制。和我们在本书中介绍的任何工具一样，数据采集工具是对人类知识和直觉的补充，而不是代替。没有哪个程序能接受一堆原始数据并生成有意义的结果，更不会生成符合你所选择期刊出版质量要求的论文。正如我们在前面强调的，提出有意义的问题、试验设计和对结果进行有意义的解释是你自己的责任。

数据采集和机器学习中的问题

数据采集涉及的是统计学家和应用数学家研究了几十年的问题。因而，统计学和数据采集之间的界限是模糊的。如果你的工作需要使用数据采集或机器学习技术，就不能只停留在精通传统统计技术上。如果你的问题可以通过最新的数据采集算法或直接的统计计算来解决，选择简单计算就比较合适。同样，要避免还未先查阅统计学书籍看是否已经有了合适的方法就设计你自己的计算方法。一般情况下，选择标准方法要比非标准方法容易调试，而且容易向你的同行解释。

有监督学习和无监督学习

机器学习方法大致可分为有监督和无监督的学习。有监督（supervised）学习是指向学习算法提供一套用于学习的标记的例子（训练组），然后用一套未标记的例子进行测试（测试组）。无监督（unsupervised）学习是在数据有效时执行的，但并不知道每个例子的正确标记。对数据运行学习算法的目的是发现一些有助于理解数据的模式或趋势。如第八章中介绍的 MEME程序，为了在一组未比对的序列中发现序列基序而执行无监督学习。事先并不知道每个序列中是否包含某一模式及其位置和外观。

聚类分析（cluster analysis）是另一种无监督学习，它在微阵列数据分析方面受到关注。聚类，如图 14-5所示，是指事先不知道数据类型时进行数据分类的过程，使相似条目在同一类中，而不相似条目在不同的类中。图 14-5显示了两个画面，左边是一些简单的点，右边的点由组的边界包围。

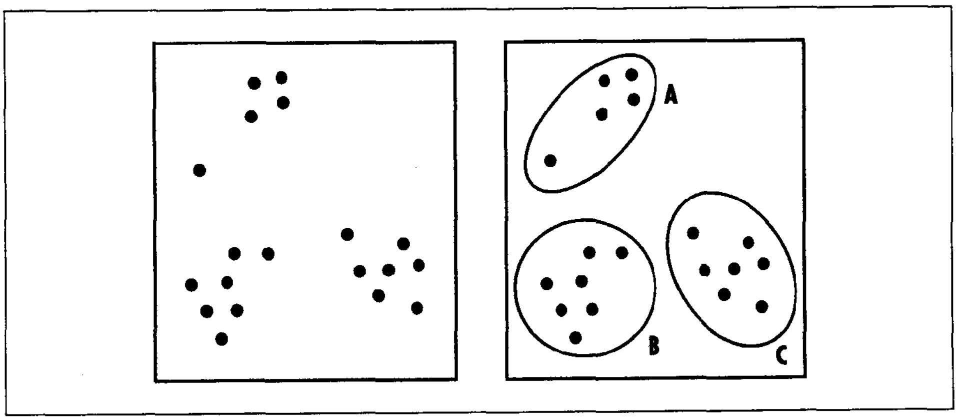


图 14-5　聚类

数据采集技术

本节我们将介绍一些常在生物信息学文献中报道的数据采集方法本节的目的是提供数据分析的复杂技巧的一个小结，并不期望你能在你选择的编程语言中实现这些算法。不过，如果你在一篇论文中看到任何一种方法用来分析数据，你应该能识别，而且如果有必要，评价其应用的方式。和实验生物学的任何技术一样，掌握计算生物学中使用的机器学习方法对于确定它们是否被恰当而正确地使用非常重要。

决策树

决策树最简单的形式是一列答案为是或否、分层排列、产生决策的问题。例如，确定一段 DNA是不是一个基因，我们可能就有一个如图 14-6所示的决策树。

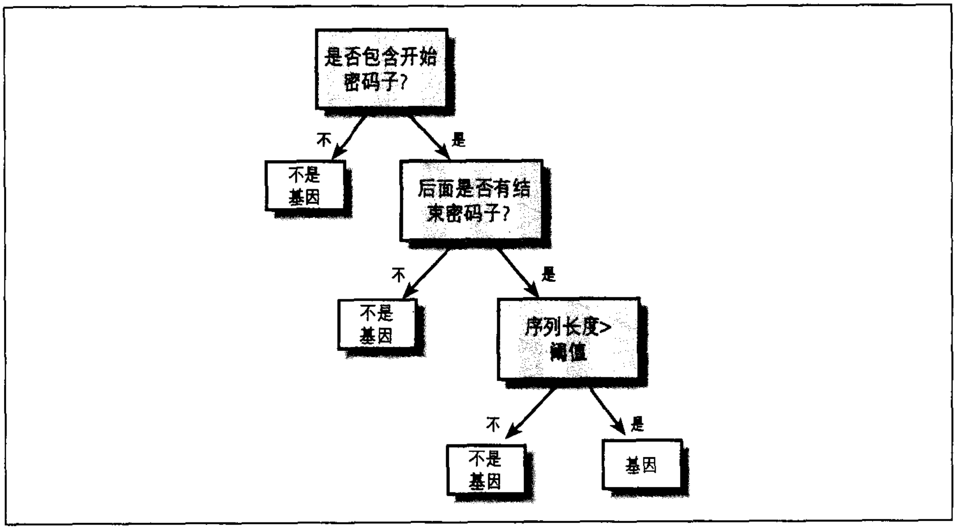


图 14-6　简单基因决策树

这样的一个决策树是很容易完成的，因为它在每个分支处都有有限种可能性，而且树中的任何途径都能产生一种决策。树的结构和每个分支的规则是学习算法根据数据决定的。Leo Breiman及其合作者在 20 世纪 80年代初期提出了学习决策树技术，其后由 J. R. Quinlan 在机器学习领域推广，他的免费 C4.5 决策树软件和商业后继方案 C5 是这个领域的标准。

与其他机器学习技术相比，决策树的一个主要优点是它们生成的是人类可以解释的模式。这是一个重要的性质，专家可以查看决策树得到的一组规则，并确定在真实条件下得到的是不是似是而非的模式（注 5）。在生物学中，树形分类程序多用于模式识别问题，例如寻找基因剪切位点或鉴定新的蛋白质家族成员。 Steven Salzberg 等开发的 MORGAN 基因查找程序就是决策树方法寻找基因的一个例子。

神经网络

神经网络(neural network ）是用于模式识别和分类的统计模型。神经网络最初是在 20 世纪 40 年代开发的一种记忆数学模型，由于它们表现为由加权函数连接的节点（通常是变量），因此有时也被称为连接模型（connectionist model）。图 14-7 所示的就是形成神经网络的过程。请注意，“神经”不特指这些模型的任何部分，也不存在实际的物理节点和连接。神经网络的含义是，通过共同的工作，这些简单的处理部件可以完成复杂的计算。

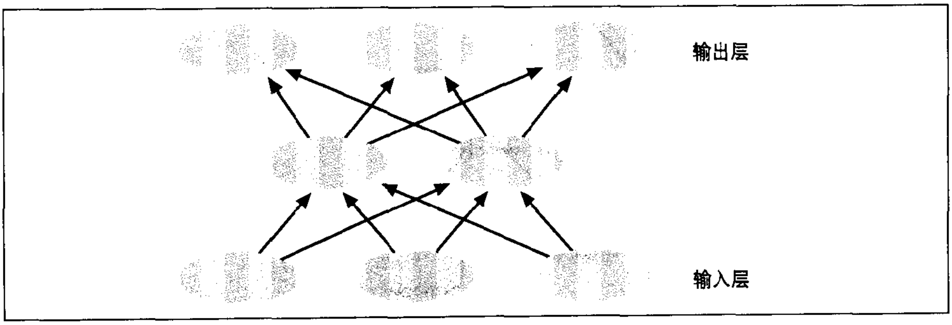


图 14-7 神经网络图表框

神经网络是由一组以规定的拓扑结构连接的节点组成的，每个节点都有到其他节点的输入和输出连接。通常，神经网络得到一个输入（例如，需要预测二级结构的一个氨基酸序列），设置第一层（输入层）节点的值。这些值通过转移函数（连接）传递给下一层节点，直至输出层。输出层启动的模式就是网络的输出结果。神经网络广泛地应用在生物信息学中，例如，第九章介绍的二级结构预测程序 PHD（*http://*

注 5： 规范决策树据说来自一家长途电话公司用于了解用户流失到其他长途电话公司的过程（churn）所使用的决策树。他们发现流失的大部分顾客是 70 岁以上的用户。人们认识到一个程序没有认识到的问题：人会死于年老。因此，能解释结果可能会有帮助。

www.embl-heidelberg.de/predictprotein/ predictprotein.html ）和 PSIPRED（*http://insulin.brunel.ac.uk/psipred*），以及第七章介绍的 GRAIL 基因查找程序。

遗传算法

遗传算法是优化算法。它们在众多方案中选取最好的一个，"最好的"是由成本函数（cost function）或适合度函数（fitness function）确定的。和神经网络一样，这些模型是从生物学概念（人类遗传学）中得到的灵感，但并没有内在的生物学性质。在遗传算法中，随机产生若干候选方案。把这些候选方案像染色体一样编码。然后每个染色体的各部分按同源重组的方式与真的染色体之间交换。最后得到的重组策略按照适合度函数计算，最高得分的染色体传递给下一阶段。重组和传递的循环继续下去，直至找到最合适的方案。遗传算法常用于分子模拟，如蛋白质的剪切和折叠。

SVM

1998 年末，一种称为 SVM（Support Vector Machine，支撑矢量机器）的机器学习工具引起了生物信息学领域的极大关注。它是由机器学习方法的先驱，Bell实验室的 Vladimir Vapnik开发的，广泛应用在从光学字符识别到金融时序分析和识别垃圾邮件等各种问题上。首先把 SVM应用于生物学的是 UC Santa Cruz 的 Tommi Jaakola（现在供职于 MIT）、David Haussler 和他们的同事，他们把SVM 用于蛋白质序列分类。从此 SVM被应用在标准计算生物学的许多难题（功能预测、结构预测和基因查找）中，其中在微阵列分析方面的应用最受瞩目。

SVM是设法找到一个多维空间中不同类型的点之间的线性分隔的有监督分类器。在二维空间中，这个分隔符是一条直线；在三维空间中则是一个平面。通常，这个分隔界面称为超平面（hyperplane）。SVM 有两个特殊的性质：第一，它不仅可找到分隔超平面，而且保证找到最优分隔超平面，或超平面产生的两个类型之间的分隔最大。离两个类型边缘最近的数据点称为支撑矢量（support vector，注6）。第二，尽管SVM是线性分类器，但它能通过把原来的数据点转换到可以用线性界面分隔的多维空间中，来对非线性分割的点进行分类。

注 6: 在这里，矢量是指数据点的坐标。例如，在一个二维图上，我们可能会用坐标对（x，y）表示数据点的位置。这些有序对就是矢量。

表 14-2列出了最常用的数据采集工具和技术。

表 14-2　数据采集工具和技术

目的	原因	可用工具
聚类	事先不知道分类方案时找到相似条目	聚类算法、自组织作图
分类	根据分类表标记每个数据	决策树、神经网络 、SVM
回归	从一些例子外推趋势	回归算法、神经网络、SVM、决策树
组合推算	提高预测的可信度	投 票(voting)方 法、混合方法

参考文献

Unix

B.McCarty. *Learning Red Hat Linux*. O'Reilly & Associates. B.McCarty. *Learning Debian GNU Linux*。O'Reilly & Associates 。利用 Linux资源创建各种系统的很好的介绍性指南。

J. Peek G. Todino，J. Strang. *Learning the Unix Operation System*. O'Reilly & Associates。适用于初学者的关于 Unix 的简明介绍。

S. Hawkins，J. Brockmeier。*The Linux Desk Reference* . Prentice Hall

Siever *Linux in a Nutshell*。O'Reilly & Associates （译注 1）。Linux 指令的快速指南。

M. Welsh，L. Kaufman *Running Linux*. O'Reilly & Associates。建立 Linux 系统的全面指南。

P.Abrahams B. Larson *Unix for the Impatient*. Addison Wesley。Unix用户需要知道的所有一切的详细而友好的演示。

译注 1：本书中文版《Linux 技术手册》即将由中国电力出版社引进出版。详情请访问：*http://www.infopower.com.cn*

A. Robbins. *Unix in a Nutshell.* O'Reilly & Associates （译注 2）。Unix指令的快速指南。

SysAdmin

A. Frisch. *Essential System Administration.* O'Reilly & Associates.Unix系统管理的详尽指南。

P. DuBois. *Using csh & tcsh.* O'Reilly & Associates 。最常用的两种 shell环境的详尽指南。

Perl

A. L. Johnson. *Elements of Programming in Perl.* Manning 。把Perl作为第一编程语言的很棒的简介。

R. Schwartz, T. Christiansen. *Learning Perl.* O'Reilly & Associates 。向有其他语言编程经验的程序员介绍如何使用 Perl。

关于更加详尽的偏重于生物学的 Perl指南，我们推荐 Cold Spring Harbor Labs 的 Lincoln Stein 实验室主页上的一个链接（*http://stein.cshl.org*）。

J. Orwant, J. Hietaniemi, J. Macdonald. *Mastering Algorithms in Perl.* O'Reilly & Associates。这一本书和下一本都包括可以使用 Perl完成的有趣的内容。

T. Christiansen, N. Torkington. *Perl Cookbook.* O'Reilly & Associates。

L. Wall, T. Christiansen, J. Orwant. *Programming Perl.* O'Reilly & Associates。Perl的圣经（译注 3）。

译注 2: 本书中文版《Unix 技术手册》即将由中国电力出版社引进出版。

译注 3: 本书中文版《Perl 语言编程》已由中国电力出版社引进出版。

一般性参考文献

R. Belew. *Finding Out About: Search Engine Technology from a Cognitive Perspective.* Cambridge University Press. 从认知科学的角度，引人入胜地讨论了信息检索和基于 Web 研究的过程。包括实践和理论两个方面。

下面的三本书包括编程的一般技术：

S. McConnell . *Code Complete*. Microsoft。

B. W. Kernighan，R. Pike. *The Practice of Programming* Addison Wesley.

J. Bentley *Programming Pearls* Addison Wesley。

生物信息学参考文献

P. Baldi, S. Brunak *Bioinformatics: A Machine Learning Approach* MIT 。本书的作者在使用神经网络、序列分析隐藏 Markov模型方面有第一手经验，包括基因查找、DNA 特征检测、蛋白质家族建模等。

A. D. Baxevanis，B. F. F. Ouellette. *Bioinformatics: A Practical Guide to the Analysis of Genes and Proteins*. John Wiley & Sons。介绍Web上的生物信息学及其工具，主要着眼于 NCBI工具。

R. Durbin, et al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press 。严谨地展示序列分析方法基本的统计和算法，包括成对和多序列分析、基序探寻和系统发育分析。

D. M. Hillis，C. Moritz，B. K. Mable，et al. *Molecular Systematics. Sinauer and Associates*。尽管前三分之二的篇幅着眼于介绍实验方法，但介绍推测方法和进化树应用的章节是 Graur 和 Li 的书的全面而严谨的继续。

生物学和分子生物学的参考文献

D. Graur, W-H. Li. *Fundamentals of Molecular Evolution. Sinauer and Associates*。

基因组随时间变化机制的可读性极强的解释，该书还讨论了基于分子数据的系统发育的推测。

D. M. Hillis, C. Moritz, B. K. Mable, et al. *Molecular Systematics*. Sinauer and Associates。尽管前三分之二的篇幅着眼于介绍实验方法，但介绍推测方法和进化树应用的章节是 Graur 和 Li 的书的全面而严谨的继续。

蛋白质结构和生物物理学

J. Israelachvili. *Intermolecular and Surface Forces*. Academic Press。大分子结构和分子生物物理学高年级学生的必备书，本书详尽介绍了分子之间、分子和界面之间的物理化学反应。

Branden, J. Tooze. *Introduction to Protein Structure*. vol1. Garland Publishing。蛋白质结构和建模基本原则的图解指南。

基因组学

T. A. Brown. *Genomes*. Wiley-Liss。从基因组学角度全面展示分子遗传学。

C. R. Cantor, C. L. Smith. *Genomics: The Science and Technology Behind the Human Genome Project*. John Wiley & Sons。如果想了解基因组学序列数据获得的细节，这是本好书。本书极其详尽地介绍了测序和制图的实验过程，并探讨了序列分析技术的未来。

生物技术学

M. Schena, Eds. *DNA Microarrays: A Practical Approach*. Oxford University Press。DNA 微阵列技术及其应用基础简介。

M. R. Wilkins, et al. *Proteome Research: New Frontiers in Functional Genomics*. Springer。简介 2D-PAGE 到 MALDI-TOF 等蛋白质鉴定和分析新技术。

数据库

S. Guelich, S. Gundavaram, G. Birznieks. *CGI Programming with Perl*. O'Reilly & Associates。介绍用来生成活动内容网页的 CGI,如果对Web软件开发感兴趣的话，这本书是个好的起点。

J. Celko. *Joe Celko's Data and Databases: Concepts in Practice*. Morgan Kaufman。介绍关系数据库概念和 SQL 的使用。

P. DuBois. *MySQL*. New Riders。MySQL 使用指南，详尽阐述了管理和安全问题。

R. J. Yarger, G. Reese, T. King. *MySQL & mSQL*. O'Reilly 。介绍 MySQL 和 mSQL 的使用，还介绍了 RDB 概念和数据库正规化。如果想从使用 Oracle 开始，O'Reilly 还出版了一些关于 Oracle 的书。

可视化

D. C. Hoaglin, et al. *Understanding Robust and Exploratory Data Analysis*. John Wiley & Sons。可视化技术的经典之作。本书主要着眼于人工分析技术，而不是最近的计算机技巧。描述的方法可在许多可视化软件包中实现，很容易运用于最近的生物信息学问题。

E Tufte. *The Visual Display of Quantitative Information, Envisioning Information, and Visual Explanations. Graphics Press*。Tufte利用来自报纸、广告、火车时刻表的例子，说明了可视数据分析中的各种实例。

W. Schroeder, K.Martin, B. Lorensen. The Visualization Toolkit: An Object-Oriented Approach to 3-D Graphics. Prentice Hall Computer Books。对那些想设计可视化工具的读者来说，这本书将计算机图表的介绍和可视化操作实践结合起来，其中有关于完全可视化系统——可视化工具箱（Visualization Toolkit, VTK）的实现的描述。VTK 是构建可视化工具的面向对象的脚本框架。它可以从 *http://www.kitware.com* 得到。

数据采集

I. Witten，E. Frank. *Data Mining: Practical Machine Learning*. Morgan Kaufman。数据采集方法的简明介绍。它来源于作者的 WEKA 程序套件 ——一系列用 Java写成的能从 Web 上免费下载的数据采集工具。

D. Pyle. *Data Preparation for Data Mining*. Morgan Kaufman 。读者可进一步深究数据准备过程。

T. Mitchell. *Machine Learning*. McGraw-Hill。提供了前面一本书中提到的相同方法的补充，更正式，但实践较少。

Brian D. Ripley，William N. Venables. *Modern Applied Statistics with S-Plus*。

Springer Verlag. S. A. Teukolsky ,et al. *Numerical Recipes in C*. Cambridge University Press。介绍支持数据分析的重要方法的各种技术。将数学解释和高效的C实现结合起来。除了硬拷贝的方式外，全书和源代码还可以从 *http://www.nr.com* 免费得到。

词汇表

adenine
腺嘌呤

alignment
比对

amino acid
氨基酸

angstrom
埃（长度单位）

Arabidopsis thaliana
拟南芥

bacteriophage
噬菌体

base
碱基

bioinformatics
生物信息学

biomolecule
生物分子

BLAST(Basic Local Alignment Search Tool)
基本局部比对搜索工具

catalytic site
催化位点

CGI (Common Gateway Interface)
公共网关接口

chromosome
染色体

codon
密码子

complexation
络合

connectionist model
连接模型

cost function
成本函数

covalent
共价

cytosine
胞嘧啶

daemon
守护进程

database
数据库

degenerate
简并性

degradation
降解

变性

脱氧核苷酸

dihedral angel
二面角

dipole
偶极

directory
目录

electrophoresis
电泳

entropically
熵

enzyme
酶

eukaryote
真核细胞

evolution
进化

exon
外显子

field
字段

file
文件

fingerprint
指纹

适合度函数

folder
文件夹

foreign key
外键

gap
空位

gel
凝胶

genetic code
遗传密码

genome
基因组

guanine
鸟嘌呤

hash
散列

homology
同源性

Human Genome Project
人类基因组计划

hyperplane
超平面

hypotheses
假说

identifier
标识符

inference
推理

intercellular
（细）胞间的

intron
内含子

jury decision
陪审团决定法

machine learning
机器学习（能力）

macromolecule
大分子

match
匹配

messenger
信使

metabolism
代谢

metabolomics
代谢组学

microarray
微阵列

microbe
微生物

minisequence
小序列问

mmCIF（Macromolecular Crystallographic Information File）
大分子晶体信息文件

monomer
单体

motif
基序

mutation
突变

NCBI（National Center for Biotechnology Information）
美国国家生物技术信息中心

nomenclature
命名法

nucleotide
核苷酸

oligomer
寡聚体

ORF（open reading frame）
开放阅读框

organelle
细胞器

orientation
定向

orthologs
直向同源

pair potential
偶势能

paralogs
共生同源

pathogen
病原体

pattern
模式

PDB (Protein Data Bank)
蛋白质数据银行

PDL (Perl Data Language)
Perl 数据语言

peptide
肽

phenotype
表现型

phylogenetic
系统发育

plasmid
质粒

point mutation
点突变

polymer
多聚体

precursor
前体

primary key
主键

primer
引物

profile
简图

promoter
启动子

protein
蛋白质

proteome
蛋白组

proteomics
蛋白质组学

proton acceptor
质子受体

proton donor
质子供体

query
查询

regular expression
正则表达式

relational databade
关系数据库

ribonucleotide
核苷酸

ribose
核糖

ribosomal
核糖体

salt-bridge
盐桥

segmental mutation
片段突变

shotgun
鸟枪法

sidechain
侧链

spectroscopy
光谱

splice site
剪接位点

string
字符串

substrate
底物

superposition
重合

SVM (Support Vector Machine)
支撑矢量机器

synthesis
合成

tag
标签

threading method
穿线（方）法

thymine
胸腺嘧啶

transcribe
转录

transfer
转运

translation
翻译

tuple
元组

uracil
尿嘧啶

voting
投票（方法）

XML (eXtensible Markup Language)
可扩展标记语言

zero spin
零自旋